University of Alberta

Adaptive Logic Networks in a Brain-Computer Interface System

By

Mark John Polak ©

A thesis submitted to the Faculty of Graduate Studies and Research in partial fulfillment of the requirements for the degree of Doctor of Philosophy.

Department of Computing Science

Edmonton, Alberta

Fall 2000

National Library of Canada
Acquisitions and Bibliographic Services
395 Wellington Street
Ottawa ON K1A 0N4
Canada

Bibliothèque nationale du Canada
Acquisitions et services bibliographiques
395, rue Wellington
Ottawa ON K1A 0N4
Canada

Your file Votre référence

Our file Notre référence

The author has granted a non-exclusive licence allowing the National Library of Canada to reproduce, loan, distribute or sell copies of this thesis in microform, paper or electronic formats.

The author retains ownership of the copyright in this thesis. Neither the thesis nor substantial extracts from it may be printed or otherwise reproduced without the author's permission.

L'auteur a accordé une licence non exclusive permettant à la Bibliothèque nationale du Canada de reproduire, prêter, distribuer ou vendre des copies de cette thèse sous la forme de microfiche/film, de reproduction sur papier ou sur format électronique.

L'auteur conserve la propriété du droit d'auteur qui protège cette thèse. Ni la thèse ni des extraits substantiels de celle-ci ne doivent être imprimés ou autrement reproduits sans son autorisation.

0-612-59652-4

Canada

University of Alberta

Library Release Form

Name of Author: Mark John Polak

Title of Thesis: Adaptive Logic Networks in a Brain-Computer Interface System

Degree: Doctor of Philosophy

Year this Degree Granted: 2000

Permission is hereby granted to the University of Alberta Library to reproduce single copies of this thesis and to lend or sell copies for private, scholarly or scientific research purposes only.

The author reserves all other publication and other rights in association with the copyright in the thesis, and except as herein before provided, neither the thesis nor any substantial portion thereof may be printed of otherwise reproduced in any material form whatever without the author's prior written permission.

#1009, 11307 – 99 Avenue
Edmonton, AB
T5K 0H2
Canada

October 2, 2000

Abstract

A new parallel man-machine training approach to brain-computer interface (BCI) succeeded through a unique application of machine learning and pattern recognition methods. The BCI system could train users to control an animated cursor on the computer screen by voluntary electroencephalogram (EEG) modulation. Our BCI system requires only two to four electrodes, and has a relatively short training time for both the user and the machine. Moving the cursor in one dimension, our subjects were able to hit 100% of randomly selected targets, while in two dimensions, accuracies of approximately 86% and 62% were averaged in the last three sessions with our two subjects.

University of Alberta

Faculty of Graduate Studies and Research

The undersigned certify that they have read, and recommend to the Faculty of Graduate Studies and Research for acceptance, a thesis entitled **Adaptive Logic Networks in a Brain-Computer Interface System** submitted by Mark John Polak in partial fulfillment of the requirements for the degree of Doctor of Philosophy.

Dr. William W. Armstrong

Dr. Jonathan R. Wolpaw

Dr. Richard B. Stein

Dr. Horacio J. Marquez

Dr. Hong Zhang

Date : 25 SEP 2000

Acknowledgement

I would like to state my appreciation to Dr. William Armstrong, my Computing Science supervisor, for many years of excellent advice and encouragement. I would also like to express my sincere gratitude to Dr. Aleksandar Kostov from the Faculty of Rehabilitation Medicine, who passed away February 25, 2000. Dr. Kostov introduced me to brain-computer interface, and was the person principally supervising my biomedical engineering and neuroscience activities. I am grateful for all the guidance that he gave me in academics and life in general. In addition, it is my pleasure to thank Dr. Richard Stein for his help in finishing my thesis, and all of our subjects who kindly volunteered their time.

Table of Contents

INTRODUCTION 1

BACKGROUND ON BCI 5

PARALLEL HUMAN-MACHINE TRAINING IN EEG-BASED CURSOR CONTROL 10

3.1 A REAL-TIME HUMAN-MACHINE TRAINING TASK 10

3.2 SYSTEM DESCRIPTION 12

3.3 PARAMETER SELECTION 14

Electrode Selection 14

Window Size and Overlap 15

Feature Extraction 17

Feature Vector Buffering 18

Adaptive Classifier 19

COMPARISON OF FAST FOURIER TRANSFORM AND AUTOREGRESSIVE FEATURE EXTRACTION METHODS 20

4.1 FEATURE EXTRACTION IN BCI 20

4.2 METHODS 26

Experimental Setup 26

Subject Training 26

Off-Line Analysis 28

4.3 RESULTS 31

4.4 CONCLUSIONS FOR FEATURE EXTRACTION 35

CLASSIFICATION 39

5.1 EXPERIMENTAL SETUP 39
Randomness in Adaptive Classifiers 39
Training and validation sets 40
5.2 CLASSIFIERS 42
Adaptive Logic Network 42
K-Nearest Neighbor 50
Classification Tree 55
Linear and Quadratic Discriminant Analysis 57
Multi-Layer Perceptron with Back Propagation learning law 61
Learning Vector Quantization 67
5.3 DISCUSSION OF CLASSIFIER PERFORMANCE 76
5.4 ENSEMBLES OF CLASSIFIERS 80
RESULTS 84
6.1 ONE DIMENSIONAL CURSOR CONTROL 84
Subjects 84
Conclusions from One-Dimensional Control Sessions 85
6.2 TWO DIMENSIONAL CURSOR CONTROL RESULTS 88
BCI APPLICATIONS 90
7.1 TRAINING SETUP FOR CONTROL OF NEURAL PROSTHESIS 90
Introduction 91
Methods 91
Results 91
Discussion 92
7.2 ENVIRONMENTAL CONTROL BY A BRAIN-COMPUTER INTERFACE 92

Environmental Control Background....92
Methods....93
Results....95
Discussion....96
CONCLUSION....98
APPENDIX....103

List of Tables

Table 5-1. Number of samples of each class in the data sets for subject L.G. 40
Table 5-2. Number of samples of each class in the data sets for subject F.K. 40
Table 5-3. Confusion matrix for ALN with K=2 on the 16 feature data for subject L.G. 48
Table 5-4. Confusion matrix for ALN with K=2 on the 32 feature data for subject L.G. 49
Table 5-5. Confusion matrix for ALN with K=2 on the 16 feature data for subject F.K. 49
Table 5-6. Confusion matrix for ALN with K=2 on the 32 feature data for subject F.K. 50
Table 5-7. Confusion matrix for 128-NN on the 32 feature data with subject L.G. 53
Table 5-8. Confusion matrix for 128-NN on the 16 feature data with subject L.G. 53
Table 5-9. Confusion matrix for 128-NN on the 16 feature data with subject F.K. 54
Table 5-10. Confusion matrix for 128-NN on the 32 feature data with subject F.K. 54
Table 5-11. Confusion matrix for C-tree on the 16 feature data for subject F.K. 56
Table 5-12. Confusion matrix for C-tree on the 32 feature data for subject L.G. 56
Table 5-13. Confusion matrix for C-tree on the 16 feature data for subject L.G. 56
Table 5-14. Confusion matrix for C-tree on the 32 feature data for subject F.K. 56
Table 5-15. Confusion matrix for LDA on the 16 feature data for subject L.G. 58
Table 5-16. Confusion matrix for QDA on the 16 feature data for subject L.G. 58
Table 5-17. Confusion matrix for LDA on the 32 feature data for subject L.G. 58
Table 5-18. Confusion matrix for QDA on the 32 feature data for subject L.G. 59
Table 5-19. Confusion matrix for LDA on the 16 feature data for subject F.K. 59
Table 5-20. Confusion matrix for QDA on the 16 feature data for subject F.K. 59
Table 5-21. Confusion matrix for LDA on the 32 feature data for subject F.K. 60
Table 5-22. Confusion matrix for QDA on the 32 feature data for subject F.K. 60
Table 5-23. Confusion matrix for MLP-BP on the 16 feature data set with subject L.G. 65
Table 5-24. Confusion matrix for MLP-BP on the 32 feature data set with subject L.G. 66
Table 5-25. Confusion matrix for MLP-BP on the 16 feature data with subject F.K. 66
Table 5-26. Confusion matrix for MLP-BP on the 32 feature data with subject F.K. 67
Table 5-27. Confusion matrix for LVQ1 on the 16 feature data. Subject = L.G. 70
Table 5-28. Confusion matrix for LVQ2 on the 16 feature data. Subject = L.G. 71
Table 5-29. Confusion matrix for LVQ3 on the 16 feature data. Subject = L.G. 71
Table 5-30. Confusion matrix for LVQ1 on the 32 feature test set. Subject = L.G. 71
Table 5-31. Confusion matrix for LVQ3 on the 32 feature data. Subject = L.G. 72
Table 5-32. Confusion matrix for LVQ2 on the 32 feature test set. Subject = L.G. 72
Table 5-33. Confusion matrix for LVQ1 on the 16 feature data. Subject = F.K. 74
Table 5-34. Confusion matrix for LVQ2 on the 16 feature data. Subject = F.K. 74
Table 5-35. Confusion matrix for LVQ3 on the 16 feature data. Subject = F.K. 74
Table 5-36. Confusion matrix for LVQ1 on the 32 feature data. Subject = F.K. 75

Table 5-37. Confusion matrix for LVQ3 on the 32 feature data. Subject = F.K. 75
Table 5-38. Confusion matrix for LVQ2 on the 32 feature data. Subject = F.K. 75
Table 7-1. Preliminary Results From Subject Trials 96

List of Figures

Figure 2-1. Averaged power spectrum density for UP and DOWN patterns. 6
Figure 2-2. Graz BCI outline 7
Figure 3-1. An example of the user's screen during an on-line experiment. 10
Figure 3-2. Data flow diagram of the real-time man-machine learning system. 11
Figure 3-3. International 10-20 system 14
Figure 3-4. Relationship between window size and the PMC 15
Figure 3-5. Overlapping EEG windows used for feature extraction. 16
Figure 3-6. Relationship between window size, AR order, and the PMC. 17
Figure 3-7. Relationship between vector buffer size and the PMC. 18
Figure 4-1: Power approximations of a 12 Hz sine wave added to a 20 Hz wave 24
Figure 4-2. Accuracy achieved during on-line evaluation by subject L.G. 25
Figure 4-3. The feature extraction and classification process used in off-line analysis. 28
Figure 4-4. Average FFT modeled spectrum for the evaluation data in a sample session. 29
Figure 4-5. AR (32nd order) modeled spectrum 30
Figure 4-6. Average values of AR coefficients for the evaluation data of one session. 31
Figure 4-7. Using three methods of feature extraction at different numbers of features 33
Figure 4-8. The duration of training per session for subject L.G. 34
Figure 4-9. Using the three methods of feature extraction at the best number of features. 36
Figure 4-10. A comparison of the three feature extraction methods by three subjects. 37
Figure 4-11. Average PMC in off-line analysis of the three feature extraction methods. 38
Figure 5-1. Structure of MAX and MIN operators, and linear pieces. 42
Figure 5-2. A simple example of an ALN approximated function. 43
Figure 5-3. Value of K in tolerances vs. PMC for subject L.G. at 16 input features. 47
Figure 5-4. Value of K in tolerances vs. PMC for subject L.G. at 32 input features. 48
Figure 5-5. NN vs. PMC for the K-NN method with 16 feature data. Subject = L.G. 51
Figure 5-6. NN vs. PMC for the K-NN method with 32 feature data. Subject = L.G. 52
Figure 5-7. NN vs. PMC for the K-NN method with 16 feature data. Subject = F.K. 52
Figure 5-8. NN vs. PMC for the K-NN method with 32 feature data. Subject = F.K. 53
Figure 5-9. Example of a two class linear discriminant. 57
Figure 5-10. Diagram of abstract neuron model. 61
Figure 5-11. A simple 3-2-1 neural net architecture. 62
Figure 5-12. Number of neurons vs. PMC with 16 feature data for subject L.G. 63
Figure 5-13. Number of neurons vs. PMC with 32 feature data for subject L.G. 63
Figure 5-14. Number of neurons vs. PMC with 16 feature data for subject F.K. 64
Figure 5-15. Number of neurons vs. PMC with 32 feature data for subject F.K. 65
Figure 5-16. Codebook size vs. PMC with 16 feature data. Subject = L.G. 69

Figure 5-17. Codebook size vs. PMC with 32 feature data. Subject = L.G. 70
Figure 5-18. Codebook size vs. PMC with 16 feature data. Subject = F.K. 73
Figure 5-19. Codebook size vs. PMC with 32 feature data. Subject = F.K. 73
Figure 5-20. PMC with different classifiers on the 16-feature data set for subject L.G. 77
Figure 5-21. PMC with different classifiers on the 32-feature data set for subject L.G. 78
Figure 5-22. PMC with different classifiers on the 16-feature data set for subject F.K. 79
Figure 5-23. PMC with different classifiers on the 32-feature data set for subject F.K. 79
Figure 5-24. "One ALN", "Voting", and "Bagging&Voting" techniques. Subject = L.G. 82
Figure 5-25. "One ALN", "Voting", and "Bagging&Voting" techniques. Subject = F.K. 83
Figure 6-1. One-dimensional real-time evaluation results for J.K. 86
Figure 6-2. One-dimensional real-time evaluation results for L.U. 86
Figure 6-3. One-dimensional real-time evaluation results for R.J. 87
Figure 6-4. One-dimensional real-time evaluation results for G.M. 87
Figure 6-5. One-dimensional real-time evaluation results for L.G. 88
Figure 6-6. Two-dimensional real-time evaluation results for G.M. 89
Figure 6-7. Two-dimensional real-time evaluation results for L.G. 89
Figure 7-1. Picture of the animated wrist with the desired direction of movement. 91
Figure 7-2. A submenu of ECBCI. 93
Figure 7-3. Lock-mode. 94
Figure 7-4. The menu structure developed for evaluation. 94
Figure 7-5. ECBCI system configuration 95
Figure A-1. Averaged spectra in the testing phase of a session for subject J.K. 104
Figure A-2. Averaged spectra in the testing phase of a session for subject L.U. 105
Figure A-3. Averaged spectra in the testing phase of a session for subject R.J. 106
Figure A-4. Averaged spectra in the testing phase of a session for subject F.K. 107
Figure A-5. Averaged spectra in the testing phase of a session for subject G.M. 108
Figure A-6. Averaged spectra in the testing phase of a session for subject L.G. 109
Figure A-7. Averaged spectra in the 2-dimensional testing phase for subject G.M. 110
Figure A-8. Averaged spectra in the 2-dimensional testing phase for subject L.G. 111

Chapter 1

Introduction

Assistive devices are essential in enhancing the quality of life for individuals who have severe disabilities due to such conditions as amyotrophic lateral sclerosis or high level spinal cord injury; however, the effectiveness of most assistive devices is dependent on preserved residual movements or speech. Without any physical channels for control, the only alternative for these people may be in exploring indirect voluntary modulation of electrical fields resulting from neural processes in their brains. This can provide control signals for a simple interface between the user and the computer known as a Brain-Computer Interface (BCI). A BCI is a system which can bypass the normal motor output through the spine by using bioelectrical signals recorded from the brain during purely mental activity. Such a BCI must be able to classify EEG patterns on-line and can be used for control and communication. A frequently used application for evaluation of BCI is to control the movements of a cursor on a computer screen. This application requires the subject to learn how to modulate the EEG-signals voluntarily by using different thought patterns for different tasks.

In theory, the brain's activity should be observable in the spontaneous EEG. In practice, the resolution and reliability of the information detectable in the EEG is severely limited by the vast number of electrically active neuronal elements, the complex geometry of the brain and head, and the trial-to-trial variability in brain operations. This is a difficult asynchronous signal detection problem because the transient voluntary EEG modulated (VEM) potentials reside in non-stationary background EEG activity with very low signal-to-noise ratios (typically below -5dB [47]). In response to these problems, current efforts to develop EEG-based BCI technology have focused on identifying specific components of the EEG that can be easily recognized by computer and used for communication. To get a set of features that can be used for classification, it is necessary to transform the EEG signal into other domains, such as the frequency domain. What adds to the difficulty of this research is that a new subject does not know what thought patterns are going to give the best results, so initially the subject and machine are learning in parallel. The problems that

remain unsolved even with the most current and successful systems are slow training of subjects, low spatio-temporal resolution, and poor accuracy in two-dimensional control.

Currently there are two main approaches to a subject's training with BCI systems that do not require external stimuli. One approach uses μ- or sensory-motor-rhythm and/or ß rhythm signals recorded over the sensory motor cortex. These signals are sensitive to physical and imaginary movements of the extremities [1, 2]. The second approach, which is the one we took, uses wider distribution of EEG signals and more abstract thought patterns that are not movement-related [3, 4]. One example of this approach that is simple for subjects to learn and duplicate is using relaxing thoughts for one direction of cursor movement and stressful thoughts for the other. After several sessions these thoughts will be spontaneously replaced with more direct thoughts representing the subject's desire to move the cursor in one direction or the other. This approach is more natural because after the first few sessions it does not require indirect movement related thoughts to control the cursor, which may be difficult to use in practical applications. To achieve training with this approach, the training system has to include very fast feedback that will enable the subject to experiment with various thoughts and to quickly see which ones work well for a particular task.

The goal of this thesis is to describe our research in developing new man-machine training technology that will achieve simple control using various mental activities while taking advantage of a parallel learning process. The control actions that we achieved are one-dimensional and two-dimensional (UP - DOWN - LEFT - RIGHT) cursor movements on the computer screen. These actions are sufficient to control a GUI (Graphical User Interface) operating environment and previously developed assistive software. To achieve this goal in our Laboratory for Advanced Assistive Technology, we have been working on the development of an EEG recording and processing setup, and adaptive training methods that will maximize efficiency of extraction of the user's intentions. In order to make a BCI practical, the following three constraints have to be satisfied:

1. Minimize the training time for the final user. Some of the best current systems often require weeks of training before reasonable performance is achieved in one-dimensional control. Long training is usually one of the main obstacles to better acceptance of any new practical assistive system. The training time of users with

our one-dimensional control system varies individually and requires between one to five half-hour sessions.

2. Use as few EEG electrodes as possible. A brain-computer interface with too many electrodes becomes costly, cumbersome, and less feasible for future implantation of electrodes under the scalp. Our system requires two to four electrodes.
3. Achieve sufficiently high accuracy to provide a reliable interface between the user and the computer. With our approach, we have been able to train people to control a cursor on the screen in one and two dimensions. In one-dimensional control, the accuracy is high enough so that the user can consistently hit 100% of randomly selected targets on the screen. In two-dimensional control, the accuracy is lower. The average accuracies over the last three sessions with our two subjects were 86% and 62%.

In this thesis we present our implementation of a unique approach to BCI that uses real-time adaptive neural network training and evaluation to train both the human and machine in parallel. Chapter 2 of this thesis gives a brief overview of the BCI field in general. Chapter 3 gives a detailed description of our new parallel human-machine training paradigm. Chapter 4 describes the two most common methods of feature extraction in BCI. It shows analysis of each method with data that was recorded in our laboratory, and selects the most appropriate feature extraction method for our system. A thorough analysis of these two methods in BCI has not been previously done. Chapter 5 does a unique off-line comparison between the accuracy of various classifiers on BCI data. Comparison between a large representation of classifiers on BCI data has not been previously done, but it was necessary in order to optimize the pattern recognition component of the overall setup, and show that the Adaptive Logic Network (ALN) is the most practical pattern recognition system for this task. Chapter 6 gives the on-line results that were achieved with our one-dimensional and two-dimensional BCI systems, and chapter 7 describes the useful applications that were developed and tested using this BCI approach.

The subjects who were used in the studies described in this thesis were volunteers and were not paid for their time. Using volunteers has the advantage of working with subjects who are motivated by the desire to help scientific investigation. On the other hand, it becomes impossible to have an intense or even regular experimental timetable and the

subjects are free to leave the study at any time. For this reason, we were not able to train our subjects in controlling their EEG to their full potential and had to rely more on machine learning.

We succeeded in creating a new approach to BCI training through our distinctive application of pattern recognition methods. We were able to use this BCI system to train users to control a cursor on the screen and run some simple applications.

Chapter 2

Background on BCI

The concept of using brain electrical activity for communication and control has a long history; however, its practical realization has recently become much more promising, due to advances in computing, signal processing and pattern recognition technology. In general, the efforts to build a brain-computer interface using non-invasive EEG readings as control signals fall into two classes [5]:

1. those that use Voluntarily Modulated Spontaneous EEG (VMSE), and
2. those that use evoked potentials related to an external stimulus (Event Related Potentials or ERPs).

The VMSE approach is based on searching for specific components in the EEG signal which are endogenous or spontaneous, and which can be manipulated voluntarily by trained mental activity. In one variant, the amplitude of ongoing brain electrical activity, recorded over specific regions of the brain, is estimated in specific frequency bands by performing power spectrum analysis of recorded EEG signals [31]. ERP techniques, in contrast, depend mainly on the electrophysiological response triggered by external stimuli [6][7][8]. Evoked responses are characterized by their latency, amplitude and location. They are usually superimposed by noise, which is significantly larger than the signal. Precise digital signal processing techniques and/or averaging of multiple responses is required to produce usable results. Each approach has its own requirements, advantages and disadvantages. The common characteristic of both approaches is that they are time consuming, which results in slow control signal discrimination and low information capacity.

The requirement of a stimulus is one of the disadvantages of the ERP approach. This feature requires that the relevant sensory modality be intact and devoted to EEG-based communication. Its advantage over the VMSE approach is that it focuses on EEG activity that occurs at a specific time or at a specific frequency. Feature extraction uses simple methods, such as measurement of peak-amplitude or latency. Advantage of the VMSE approach is that subjects can be trained to control the amplitude and distribution of specific frequency components of their EEGs and to use that as a functional communication and control tool [9]. Potentially, this can result in a larger number of control actions and a

higher discrimination speed. Obviously, the VMSE approach employs both human and machine training in achieving its results, while the ERP approach mainly uses the physiological response to a stimulus, although its results also can be affected by training [6]. For real-time communication and control applications it is important to provide the user with the possibility to express himself or herself at any time without any external stimuli present.

There are many reports suggesting that EEG activity can provide the basis for control and communication applications [10][11][8][12][13][3][1][7]. Many studies have demonstrated that humans have the ability to control a variety of EEG phenomena [14][15][11][12]. Related animal research has indicated that EEG conditioning does not require mediation by motor responses [16][17]. While these studies have not focused on producing bi-directional control of spontaneous EEG, which would be important for control and communication, they suggest that such control is possible.

One EEG component susceptible to conditioning is the μ-rhythm, 8-12 Hz activity over primary motor and sensory cortices that is attenuated by movement but not by eye opening and it is detectable in nearly all adults [30]. This pattern is also well known as Sensory-Motor Rhythm (SMR). It is thought to be an idling rhythm of the sensorimotor cortex, as the α-rhythm is an idling rhythm of the visual cortex [18]. Using the VMSE approach, Wolpaw and McFarland found that humans could learn to increase and decrease the μ-rhythm amplitude over one hemisphere and could use this control to move a cursor on a computer screen up or down quickly and accurately [3]. As our initial step, we have tested this assumption by calculating the power spectral density (PSD) of two thought patterns that could be used to discriminate the UP and DOWN intentions. A subject was asked to think relaxing thoughts for UP, and stressful

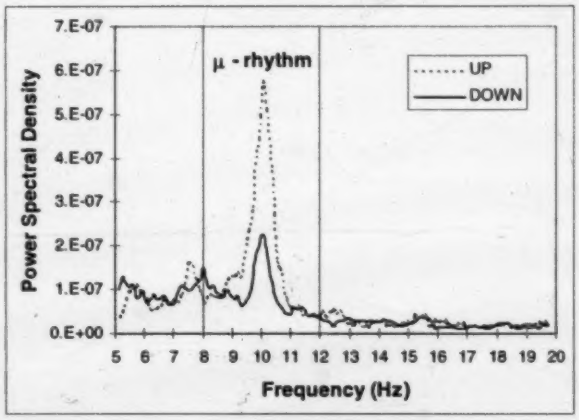

Figure 2-1. Averaged power spectrum density for UP and DOWN thought patterns in lead C4.

thoughts for DOWN while we recorded the EEG signal for two minutes for each class. Figure 2-1 shows a significant difference in μ-rhythm power which could be used for discrimination.

The real time BCI based on VEM, as described by McFarland et al. [31], uses the μ-rhythm power with linear equations to achieve one-dimensional cursor movement with above 90% success rate. They also had limited success in training subjects to generate four distinct cursor movements on the computer screen (UP - DOWN - LEFT – RIGHT) from only two EEG signals recorded over the central sulci [4][9]. In the two-dimensional case, one linear function controlled horizontal cursor movement and another controlled vertical movement. The sum of the μ-rhythm voltages over left and right sensorimotor cortices was used to control horizontal movement and their difference to control vertical movement. Autoregressive (AR) spectral estimation was used because it was found that with shorter signal segments, the frequency resolution of the AR algorithm was superior to that of the FFT method [31]. Using shorter signal segments allows a high rate of cursor movement and a rate of 8 movements per second was achieved. Non-linear classifiers, such as Distinction Sensitive Learning Vector Quantizer (DSLVQ) were applied only in off-line processing. 6 - 8 weeks of subject training was required with this system to achieve the results. The same team reported achieving 90% accuracy in being able to hit a target located at the top or bottom edge of a video screen, and up to 16 detections (cursor movements) per second with a very well trained ALS patient. For discrimination they used a linear relationship between the cursor movement and signal's amplitude with parameter values derived from evaluation of the user's previous performance. Based on these results, they suggest the applicability of VMSE based BCI to individuals

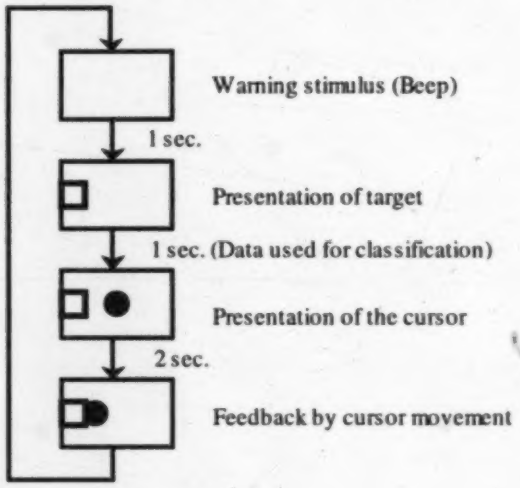

Figure 2- 2. Graz BCI outline

with brain and spinal cord injury, multiple sclerosis, cerebral palsy, Friedrich's ataxia and ALS.

Other important work done in this field is by Pfurtscheller et al. [22],[24],[25],[26],[27] who analyzed EEG during subject's preparation for hand movement to direct cursor motion on a monitor to the left or right. Their system builds on the observation that preparation of hand movement results in desynchronization of central β- and μ-rhythms [19][26]. In their experiments subjects were shown the target (left or right hand) followed by a one second pause, and then were given a cue to carry out the indicated movement, as can be seen in Figure 2.2. The one-second interval before any physical movement was used for classification. Power values in various frequency bands were calculated from leads C3 and C4. In the earlier experiments, learning vector quantization (LVQ) was selected for the classifier [27]; however, LVQ was later replaced by distinction sensitive learning vector quantization (DSLVQ) [24],[28],[29] which is an improved LVQ. One major shortcoming of standard LVQ is that all the features of the input data are treated equally. DSLVQ performs a scaling transformation to adapt the influence of different features. Applying this method they developed their BCI which discriminates preparation for movement or imagined movement of one of the index fingers. These two teams (Pfurtscheller and Wolpaw) recently joined their efforts in optimization of the EEG recording and processing methods to standardize this approach. Using multi-channel EEG recording and more sophisticated data processing methods they were able to determine the optimal electrode position for each task and for each particular subject [28].

Also using imaginary movement, Mason and Birch have been able to design an asynchronous BCI switch in attentive, spontaneous EEG. The low-frequency asynchronous switch is based on a feature set related to imaginary movements in the 1-4 Hz frequency range. Offline evaluations of a prototype switch demonstrated true positive rates in the range of 38% to 81% with corresponding false positive rates in the range of 0.3% to 11.6%. This switch should be able to augment other VMSE BCI systems that are based on frequency components in the μ- and β- rhythm range.

Regarding the ERP approach to communication and control, Farwell and Donchin [8] showed that the amplitude of the P300 peak (a late positive wave that occurs approximately 300 milliseconds after the onset of a meaningful stimulus) of the VEP was larger in response to the letters the person wished to select, and thus could be used to communicate

commands. They demonstrated that P300 can be used accurately for communication, but only at a rate of 0.20 bits/s, which resulted in 95% accuracy for 2.3 characters or messages per minute. A similar approach was adopted by Polikoff et al. with even lower accuracy [20]. Modulation of steady-state visual evoked response (SSVER) was proposed as a control and communication channel by McMillan and Calhoun [6]. Voluntary movement-related potentials (VMRP) were also investigated for a real-time human-machine interface [47]. Combination of voluntary movement-related potentials and visual evoked potentials was proposed by Patmore and Knapp [21]. They used visually evoked responses to create a feedback loop which overcomes the drift problem which frequently occurs with electrooculogram-based (EOG) cursor controllers.

Studying previously reported work brought us to the conclusion that the VMSE approach has a larger potential in the long run. It is much more difficult to start with because it involves three-way parallel learning performed at the same time by researchers, users, and the adaptive computer software. However, for efficient control and communication we believe that instead of attempting to interpret evoked EEG potentials to an external stimulus, it is more natural to follow Craggs' idea [17] and attempt to teach the user to condition his or her EEG voluntarily so that it can be easily interpreted.

BCI technology at this time still remains in its infancy. Most BCI systems are currently too slow and unrefined to be of use to most individuals. However, they can provide tremendous benefit to many individuals with disabilities for whom it is their primary means of communication.

Chapter 3

Parallel Human-Machine Training in EEG-Based Cursor Control

In this section we describe the setup for our experiments. Section 3.1 gives a high level description of the human-machine training approach and the experimental protocol. Every time instant, at intervals of 50 ms with our present hardware, the adaptive classifiers are trained and evaluated to give feedback to the subject, who at the same time is trying to learn how to modulate his/her EEG in a way that will be detectable by surface EEG. This approach is unique by having the subject and classifier train in real-time simultaneously [32], instead of in a repeatable sequence. A modular overview of the system we implemented is given in Section 3.2. The system is modular and flexible, but as a result has a large number of adjustable parameters. Parameter optimization is discussed in Section 3.3.

3.1 A Real-Time Human-Machine Training Task

The control actions that we want to achieve through the BCI are cursor movements. We chose cursor movement because it is objectively measurable, easily implemented, simple for the user to learn, and can serve as a prototype for control of a wide variety of applications. It has been shown by McFarland et al. in a similar BCI setup that feedback on performance is not necessary for EEG control [34]. Since control of voluntarily modulated EEG, unlike visual evoked potentials, is not dependent on the sensory input provided by cursor movement, other non-visual

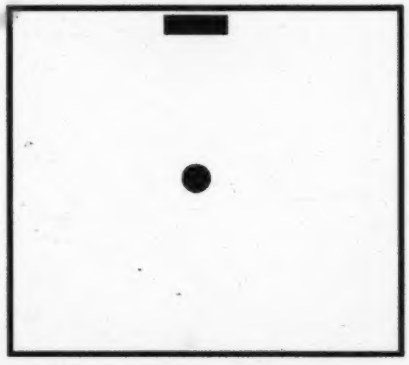

Figure 3-1. An example of the user's screen during an on-line experiment. The user's goal is to move the round cursor to the rectangular target.

forms of feedback may be possible.

In training, a target is selected as either UP or DOWN in our one-dimensional system and UP, DOWN, LEFT, or RIGHT in our two-dimensional system. The subject is asked to move the cursor that appears in the middle of the screen to this target, as can be seen in Figure 3-1. A pre-selected minimum number of cursor movements are required to

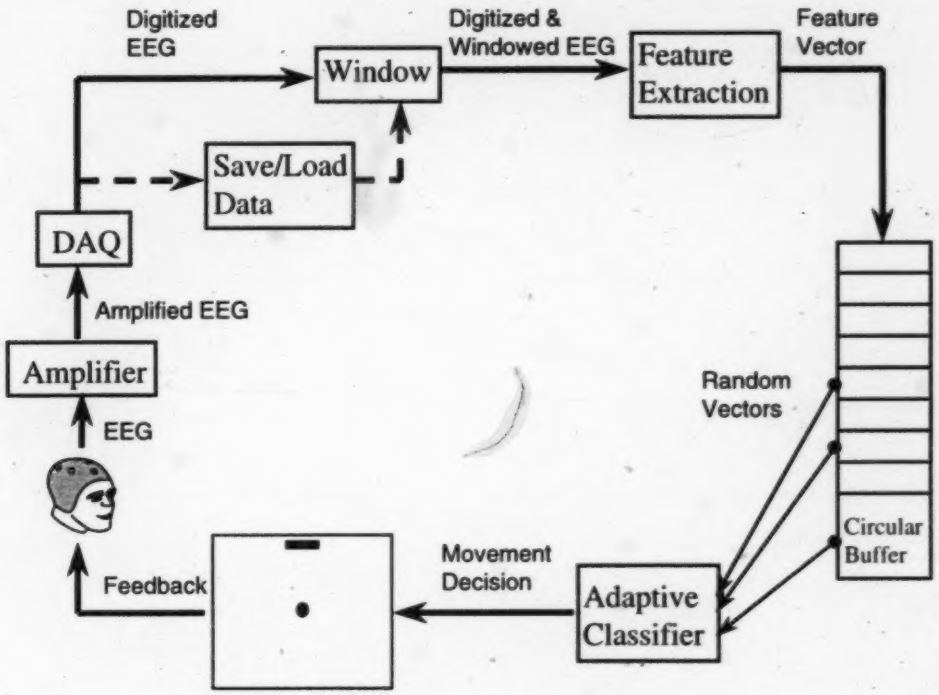

Figure 3-2. Data flow diagram of the real-time man-machine learning system.

reach the target. The subject's task is to move the cursor to the target by modulating their EEG. The trial ends when the cursor touches the target or touches the opposite edge of the screen. If it touches the target, the target turns bright green and the computer registers a hit. If it touches the opposite edge, the cursor disappears and the computer registers a miss. In either case, a brief pause ensues, followed by the next trial. During the first experiments with a subject, two learning processes occurred in parallel:

- subjects try to determine which thoughts produce desired effects on the cursor movement, and

- adaptive classifiers train in real-time to discriminate the patterns of extracted EEG features associated with desired cursor movement.

Evaluation is done at the end of each session to assess the performance of the subject and the system. In this step, training of the classifier is turned off and the performance of the system and subject are tested in a series of runs. Approximately ten minutes is required to train the system and five minutes to test it.

3.2 System Description

The real-time system that we use to simultaneously train the subject and the machine is modular so that it is possible to experiment with various feature extraction and automated machine learning algorithms. The design of this system is illustrated in Figure 3-2. The subject is comfortably seated in front of a monitor that displays the cursor movement. EEG signals are recorded using an electrode cap with up to 28 gel-filled electrodes distributed according to the 10-20 international electrode system [32], one ground electrode and the linked-ears reference. The electrode cap and EEG-preamplifiers are optically isolated from the rest of the equipment. This provides safety for both the subject and the operator. For signal amplification and filtering we use the Brain Imager (Neuroscience Inc.). Although up to 28 electrodes can be amplified and recorded at the same time, we only use two to four electrodes for the cursor control.

Analog EEG signals are then digitized at 200 samples/s by a data acquisition card (DAQ) inserted in an IBM PC compatible computer. Once the amplified EEG signals are digitized, they are saved to the hard drive for any future off-line analysis. Off-line analysis is easily facilitated by simulating the data acquisition by loading EEG samples from disk, as is shown in Figure 3-2. A selected subset of the recorded channels is used for real-time processing. Recording many channels allows extensive off-line analysis, while only carefully selected channels that are shown to carry most information are used to generate the feedback.

The digitized EEG data to be used for feedback is grouped in overlapping windows. The window size and overlap are adjustable and our choice of these parameters is discussed in the next section. Features are computed from each window of EEG. Presently we use the

autoregressive coefficients for feature extraction; however, any other method for extracting features from the EEG could possibly be used. After feature extraction, each feature vector is placed into a circular buffer (queue) according to the position of the target at that time. Each target class (UP, DOWN, LEFT, or RIGHT) has a separate buffer for storing approximately 20 seconds of past feature vectors.

Multiple randomly selected vectors are taken in equal numbers from all of the circular buffers and presented to the adaptive classifier at each cycle. Since it may take many steps to reach the target before a new target is selected, a classifier would learn only vectors for the current class. In our system, the classifier performs a training step with feature vectors selected from all the circular buffers. This solves the problem that if the training rate is high, the classifier does not become biased towards the last direction that it was trained on.

We use Adaptive Logic Networks (ALN) as our classifiers, but any real-time adaptive classification system could be used in principle. Classifiers that require the entire training set before building the classification model would not work in this real-time setup. In two-dimensional cursor movement, two classifiers are trained in parallel. One classifier is used to discriminate between the up and down vertical movement and a second classifier is used discriminate between the left and right horizontal movement. The classifier for vertical control receives feature vectors from the UP and DOWN circular buffers to train to discriminate between the two kinds of patterns. Feature vectors from the LEFT and RIGHT buffers are presented to the vertical classifier as "no-movement" class. Similarly, the classifier controlling the horizontal movement receives LEFT and RIGHT feature vectors to learn how to discriminate between these two directions and UP and DOWN vectors to learn when not to be active. Adding vectors that impede motion from buffers with movement direction perpendicular to what the classifier is trying to learn helps in reducing the interference of the two classifiers with each other. For example, when the subject is trying to move the cursor in the UP or DOWN directions, it would be undesirable to also have an uncontrollable horizontal drift of the cursor.

The outputs of the classifiers are converted into cursor movement, which is displayed on a computer monitor that the subject is watching. This completes the real-time feedback loop. With our present system, the feedback loop shown in Figure 3-2 cycles every 50 milliseconds, giving a continuous appearance to the cursor movement and training.

3.3 Parameter Selection

Electrode Selection

Up to 28 electrodes can be amplified, recorded, and used for feedback at the same time with our BCI setup; however, pattern recognition algorithms are overwhelmed by the direct input of signals of large dimension. For example, extracting only 6 features per lead would give 168 features if all 28 leads were employed for feedback. In such a high dimensional input

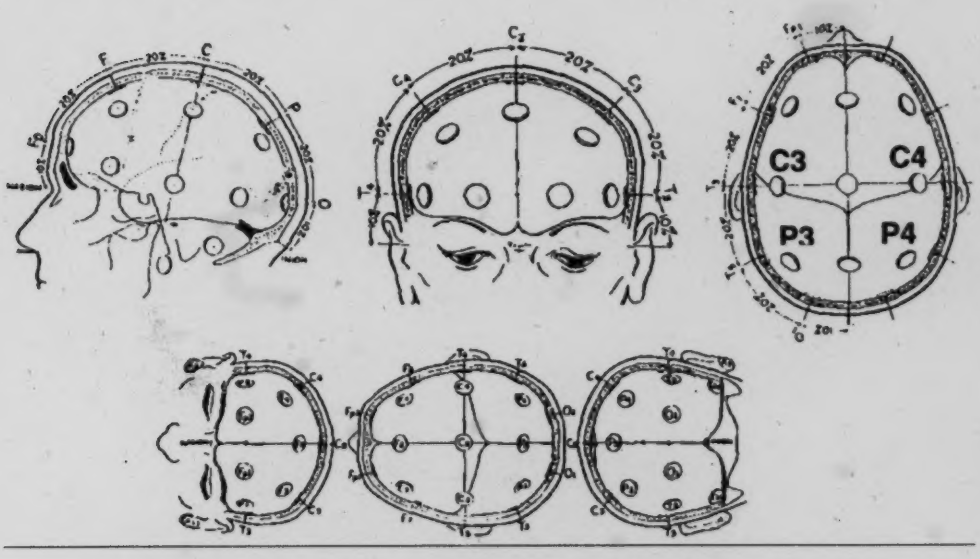

Figure 3-3. International 10-20 system showing location of leads C3, C4, P3, and P4. From: Jasper, J.J., EEG Clin. Neurophysiol., 10: 371-375, 1958 [32].

space either the accuracy of the system would suffer or else the pattern recognition system would require an extremely long training time, which is not practical when used with human subjects.

In order to minimize the number of electrodes that are employed in the feedback, we considered work done by other researchers. Both Pfurtscheller and Wolpaw only used a small number of electrodes located over the sensorimotor cortex. Most commonly, leads C3 and C4 were used [3][31][41], or a bipolar combination that could be approximated by C3-P3 and C4-P4 [25][4][23]. Position of leads C3, C4, P3, and P4 can be seen in Figure 3-3.

To test whether leads C3, C4, P3, and P4 would contain all the relevant information for BCI feedback, an experiment was carried out where the subject was trying to hit a random target at the top or bottom of the screen, and eleven leads (F3, Fz, F4, C3, Cz, C4, P3, Pz, P4, CP1 and CP2) were selected to give an adequate coverage of the sensory and motor cortices. Offline classification analysis with all eleven leads gave a probability of misclassification (PMC) of 0.349 while analysis with only leads C3, C4, P3, and P4 gave a slightly better PMC of 0.327. This result suggests that leads C3, C4, P3, and P4 contain the relevant information for classification and adding non-relevant or duplicate information only increases the input vector dimensionality and makes the classification task more difficult. Further reducing the number of leads to C3 and C4 caused an increase in PMC to 0.359, so leads C3, C4, P3, and P4 were used in most of our experiments. In a few subjects we were able to reduce the number of electrodes to just C3 and C4 without an increase in error.

Window Size and Overlap

The cycle duration at which the subject's EEG is processed and converted into feedback is adjustable. The cycle duration is set by overlapping the EEG data windows from which features are extracted. Shorter cycle duration gives feedback more often, and therefore produces smoother cursor movement but does not reduce the delay of the feedback due to the averaging effect of the window size. We chose 50 milliseconds for the cycle duration in

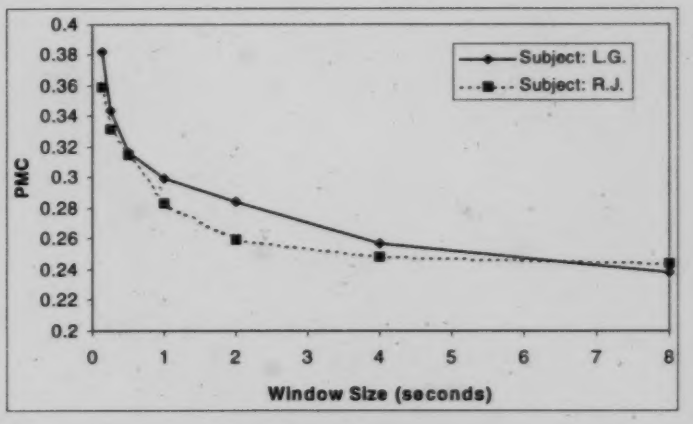

Figure 3-4. Off-line analysis of two subjects showing the relationship between window size and the PMC in a fourth order AR model. The analysis was done on three sessions for each subject and the results were averaged.

order to guarantee that our hardware (Pentium 166 MHz) is fast enough to complete all the tasks shown in Figure 3-2 with any reasonable choice of parameters.

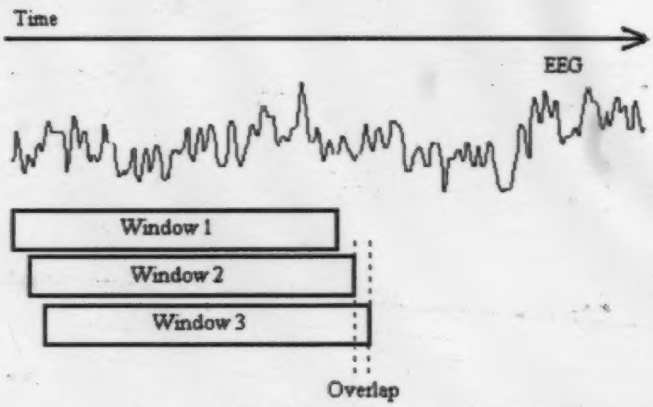

Figure 3-5. Overlapping EEG windows are used for feature extraction.

The window size determines how many consecutive EEG samples are available for feature extraction. The feature extraction module is responsible for extracting useful information from each window of EEG. Using a larger window size increases the accuracy of most feature extraction methods by providing more samples from which the features are approximated. As can be seen from off-line analysis in Figure 3-4, the relationship between the window size and the probability of misclassification (PMC) of AR coefficients calculated from each window of data is exponential. Although expanding the window size decreases the PMC of the extracted features, it increases the delay of the feedback.

As can be seen in Figure 3-5, the window overlap determines the frequency of feedback, while the window size determines how long past EEG samples affect the movement of the cursor, not taking classifier adaptation into account. When a change occurs in the EEG pattern (e.g. subject decides to change the direction of cursor movement) it will take at least one half the window size before a majority of the samples in the window are of the new pattern. Although it is not clear whether feedback to the subject is needed, when feedback is given it should be as rapid as possible. Delayed feedback degrades performance [34]. To reach a compromise in selecting window size between delay in feedback and expected PMC of each window, we chose to use a one second long window with all new subjects. As was shown in Figure 3-4, the PMC increases sharply with respect to window size as the window size is reduced below one second. The slope of PMC with

respect to window size decreases slowly for windows longer than one second. Only after a subject becomes comfortable at controlling the cursor with a one second window do we try to reduce the response time by decreasing the window size to 0.5 second and even 0.25 seconds.

Feature Extraction

Our present system calculates AR coefficients as the features to be used for classification; however, other feature extraction methods are possible. Our off-line analysis showed AR coefficients to give higher accuracy than the FFT as will be shown in Chapter 4. We experimented with various numbers of AR coefficients per lead in real time but observed that four coefficients seem to be enough for the classifier to distinguish between the classes. Figure 3-6 shows the relationship between the AR order, window size, and PMC. For a given AR order, the PMC is always lower with a larger window size. For each of the three window sizes, an AR order of two gives a relatively large PMC. The PMC decreases for AR order of four and stays relatively even for the next two AR increments, up to thirty-two, at which point the error increases again. This relationship shows that at least four AR coefficients are needed to capture the relevant information in the signal. A few extra AR coefficients do not significantly degrade performance; however, if a large enough number of

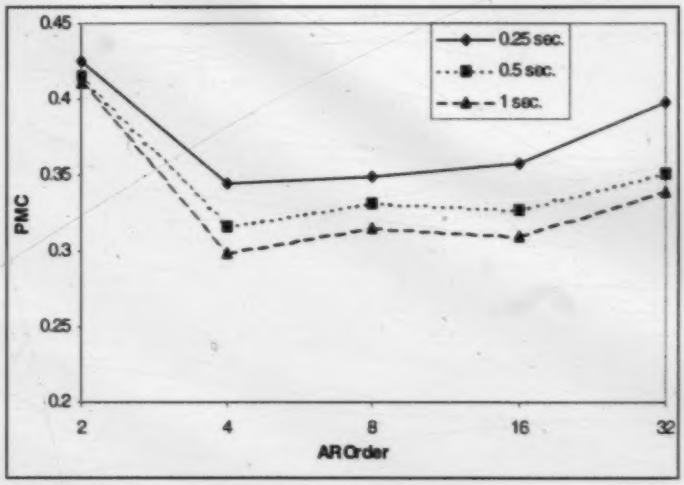

Figure 3-6. Off-line analysis showing the relationship between window size, AR order, and the PMC. The analysis was done at three window sizes (0.25 sec., 0.5 sec., and 1 sec), on three separate sessions. The results for the three sessions were averaged at each AR order and each window size.

coefficients is taken, the PMC will increase noticeably. This increase in PMC may be due to the high dimensionality and the many irrelevant features which the classifier is asked to train on.

Feature Vector Buffering

After each feature is calculated it is not immediately presented to the classifier. Instead the vector is queued in a direction dependent buffer from which the vectors are taken at random for training. The size of the vector buffer determines the duration in which each training vector may be presented to the classifier. If the vector buffer is too large then the classifier will be very slow to adapt because vectors that were generated a long time ago are still used for training. If the vector buffer is too short, then a new vector will be presented to the classifier and then quickly discarded, resulting in training that is extremely sensitive to new data. Short buffer size can be a problem if the subject momentarily loses concentration, since this would have significant influence on the classifier and can be especially damaging to the classifier if training is stopped at this time.

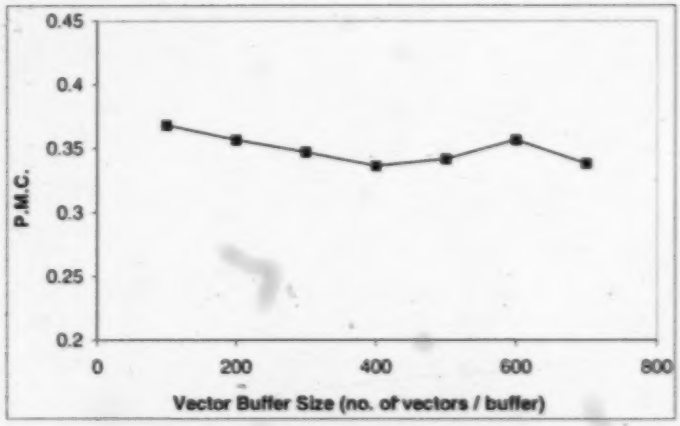

Figure 3-7. Off-line analysis showing the relationship between vector buffer size and the probability of misclassification. The analysis was done on three separate sessions each time with a window size of 1 second, and using fourth order AR model.

One way to determine an adequate vector buffer size is to assume that the buffers should always hold enough data to fully train the classifier. The classifier we presently use in our system (ALN) approximates relations with linear pieces. The number of training points required is constrained by:

$P \geq l \times (m + 1)$

where P is the number of training points required, l is the number of linear pieces in the ALN, and m is the number of inputs. The above equation gives the minimum number of training points without any consideration given to noise in the data. A more precise equation for approximating the number of required training points is:

$P = l \times (m + 1) \times v$

where v is the factor by which the number of points required by each linear piece is overdetermined. We want to make sure that at least ten linear pieces can be used to learn the function to discriminate between our UP and DOWN classes, using four channels with four input features each. Given the noisy nature of the EEG signal, let us also assume that we will require at least five times the minimum number of training points to get reasonable generalization. Using the above equation,

$P = 10 \times (16 + 1) \times 5$

we would need at least 850 training points. Since there are two data sets per classifier, this means that 425 training vectors or 21.25 seconds (425 vectors x 0.05 seconds/vector) of data needs to be kept in each buffer. If we increase the number of inputs per channel to 6 instead of 4, the buffer size would need to be 625 training vectors (31.25 seconds of recording).

Off-line analysis of the vector buffer size can be seen in Figure 3-7. The error rate is slightly higher with small buffers, but the vector buffer size does not seem to have a significant effect on off-line training.

Adaptive Classifier

The voltage potentials recorded on the scalp vary between different individuals and sometimes between sessions for the same individual. Because of the variability between EEG recordings, the classification function needs to be customized for each subject and preferably for each session. An adaptive classifier has the advantage of automatically learning to approximate the optimal classification function.

The problem of classification is of partitioning the feature space into regions, one region for each category. During training, the category into which each EEG sample belongs (UP, DOWN, LEFT, RIGHT) is always known in our application, so our case falls into supervised learning. The classifier that we use is the Adaptive Logic Network because it is the most practical solution, as is discussed in Chapter 5.

Chapter 4

Comparison of Fast Fourier Transform and Autoregressive Feature Extraction Methods

In this section of the thesis we describe the feature extraction component of our BCI system. The purpose of this is to reduce the data by extracting certain features that capture the relevant information. Perfect feature extraction would make the job of the classifier trivial, and an omnipotent classifier would not need the help of feature extraction. Unfortunately neither an all-powerful feature extractor or classifier exists, so for complex real-world problems one must use the full potential of both methods in order to get good results. In general, the problem of feature extraction is much more problem-dependent than the problem of classification. Classifiers are not originally designed as feature extractors even though they have been used as such. They are rather designed to optimize classification for given features and they can be very powerful tools if a small number of key features is supplied to them. In general, pattern recognition algorithms are overwhelmed by the direct input of raw signals of large dimension for local feature extraction. This problem is often referred to as "the curse of dimensionality" [36]. An efficient and compact representation of data suitable for the problem at hand leads to an improvement in solving the problem itself.

Section 4.1 describes the feature extraction methods commonly used with EEG analysis. Section 4.2 describes the online and offline experimental setup used in our study of feature extraction. Section 4.3 gives the results that our analysis produced, and Section 4.4 concludes the section with our observations.

4.1 Feature extraction in BCI

The first breakthrough in the automatic analysis of EEG was brought by the introduction of the FFT algorithm in 1965. The classical approach to feature extraction in BCI is to estimate the power at carefully chosen frequency bands in a FFT generated spectra [4],[37],[3],[28]. FFT is severely biased by the assumption that the signal is infinite and periodic outside the measurement window. Nevertheless, today the FFT is still the major signal processing tool used for the analysis of biomedical signals. The FFT is not the only way to estimate the

power spectrum of a process, nor is it necessarily the best way for all purposes. Parametric methods like the AR model are free from the windowing effect and give better estimates since no assumption about the signal outside the measurement window is needed. In the case of AR modeling, the signal $y(n)$ is viewed as:

$$y(n)=\sum_{k=1}^{M} a_k y(n-k)+e(n) \qquad (1)$$

where M is the order of the model, a_k are the model coefficients, and $e(n)$ is the error term. Taking the z-transform of (1) and showing it in the frequency domain gives

$$Y(z)=\sum_{n=-\infty}^{\infty} y(n)z^{-n}=\sum_{n=-\infty}^{\infty}\left[\sum_{k=1}^{M} a_k y(n-k)\right]z^{-n}+E(z)$$

where $z = e^{j2\pi f}$. Changing the order of summation, we get

$$Y(z)=\sum_{k=1}^{M} a_k\left[\sum_{n=-\infty}^{\infty} y(n-k)z^{-n}\right]+E(z)$$

Applying the relation for the z-transform of a delayed sequence, we have

$$Y(z)=\sum_{k=1}^{M} a_k z^{-k}Y(z)+E(z)$$

Since $Y(z)$ is not a function of k, it can be factored out.

$$Y(z)\left[1-\sum_{k=1}^{M} a_k z^{-k}\right]=E(z)$$

$$Y(z)=\frac{E(z)}{1-\sum_{k=1}^{M} a_k z^{-k}} \qquad (2)$$

An ideal fit of the model is assumed implying that $e(t)$, as a white noise signal, has zero mean and variance σ^2, then the power density function is

$$P(z)=\frac{\sigma^2}{\left|1-\sum_{k=1}^{M} a_k z^{-k}\right|^2} \qquad (3)$$

The most general linear filter takes a sequence of input points x_k and produces a sequence of output points by the formulae

$$y(n)=\sum_{j=0}^{N} c_j x(n-j)+\sum_{k=1}^{M} a_k y(n-k)$$

The filter response function is

$$H(z)=\frac{\sum_{j=0}^{N} c_j z^{-j}}{1-\sum_{k=1}^{M} a_k z^{-k}}$$

Equation (2) is similar to the simplest form of the general digital filter that contains only finite-valued poles and all the zeros at z=0 given by

$$H(z)=\frac{1}{1-\sum_{k=1}^{M} a_k z^{-k}}$$

The FFT power spectrum estimate for any real sampled function c_k can be written, except for normalization convention, as

$$P(f)=\left|\sum_{k=\frac{-N}{2}}^{\frac{N}{2}-1} c_k z_k\right|^2 \qquad (4)$$

which is an approximation to the "true" power spectrum (except for normalization) of

$$P(f)=\left|\sum_{k=-\infty}^{\infty} c_k z_k\right|^2$$

The FFT approximation in (4) is known as an *all-zero model* since the model spectrum can have zeros in the z-plane, but not poles. In contrast, (3) known as the all-poles model can have poles, corresponding to infinite power spectral density at real frequencies in the Nyquist interval. These poles can provide an accurate representation for power spectra that is supposed to have sharp lines. By contrast, (4) can have only zeros not poles, and must attempt to fit sharp spectral features with, essentially, a polynomial.

Although AR spectral analysis has been studied in EEG processing for a long time [38],[39],[40] it has only recently been actively applied in BCI systems [41],[34],[31],[2]. AR has two other important advantages over FFT with respect to BCI.

The first advantage of AR over FFT is that better resolution can be obtained by applying AR modeling. The AR estimation is a function of continuously varying frequency

so there is no special significance to specific equally spaced frequencies as there is in FFT. The AR estimate may have very sharp spectral features so one can evaluate it on a very fine mesh near to those features, and more coarsely farther away from them [42],[43],[44]. The second advantage of AR over FFT is that with AR, good spectral estimates can be obtained from short EEG segments [31]. Even very slow waveforms with a wavelength longer than the interval observed can be detected accurately [42]. In a BCI system, shortening the required time segment is important for reducing the delay in the feedback provided to the user. Figure 4-1 shows the spectral approximation of a combination of 12 Hz and 20 Hz sine waves that demonstrate the advantage of AR over FFT in terms of windowing effects and short window sizes.

The most serious disadvantage in AR spectral estimation is the problem of selecting the proper model order. With noisy input signals, if the order is too high, the result will be spurious peaks in the spectra; however, too low an order yields rather smooth spectra. Some experts recommend the use of the AR method in conjunction with more conservative methods, such as periodograms, to help choose the correct model order, and to avoid getting misled by spurious spectral features [43].

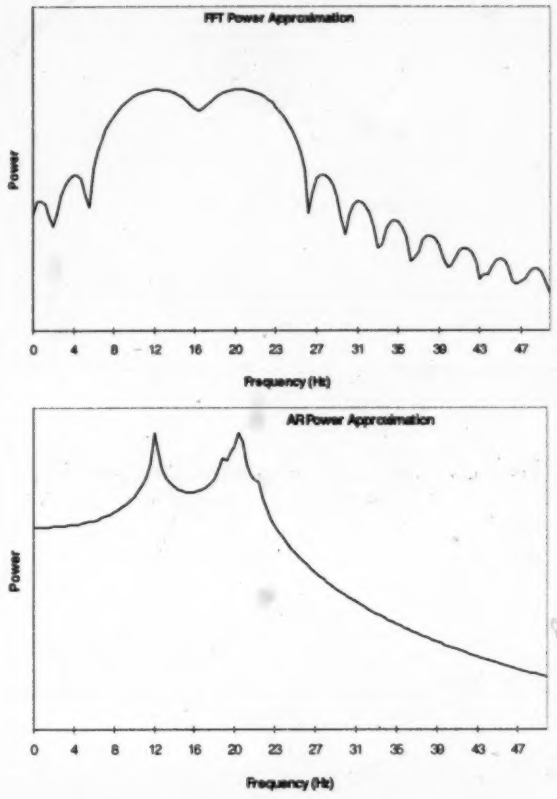
FFT Power Approximation
Power
0 4 8 12 16 20 23 27 31 35 39 43 47
Frequency (Hz)
AR Power Approximation
Power
0 4 8 12 16 20 23 27 31 35 39 43 47
Frequency (Hz)

Figure 4-1: Power approximations of a 12 Hz sine wave added to a 20 Hz sine wave sampled at 200 Hz for 0.3 seconds. Top part of the figure shows the FFT approximation, and the bottom part shows the AR model of 8th order.

In terms of operations count, calculating the AR coefficients using the Burg's method is in the order N times M, where N is the size of the signal and M is the model order, as compared to NlogN for the FFT [43]. Therefore AR is slightly slower than FFT if M is greater than logN, which may be the case in real-time BCI systems where short signal segments are used.

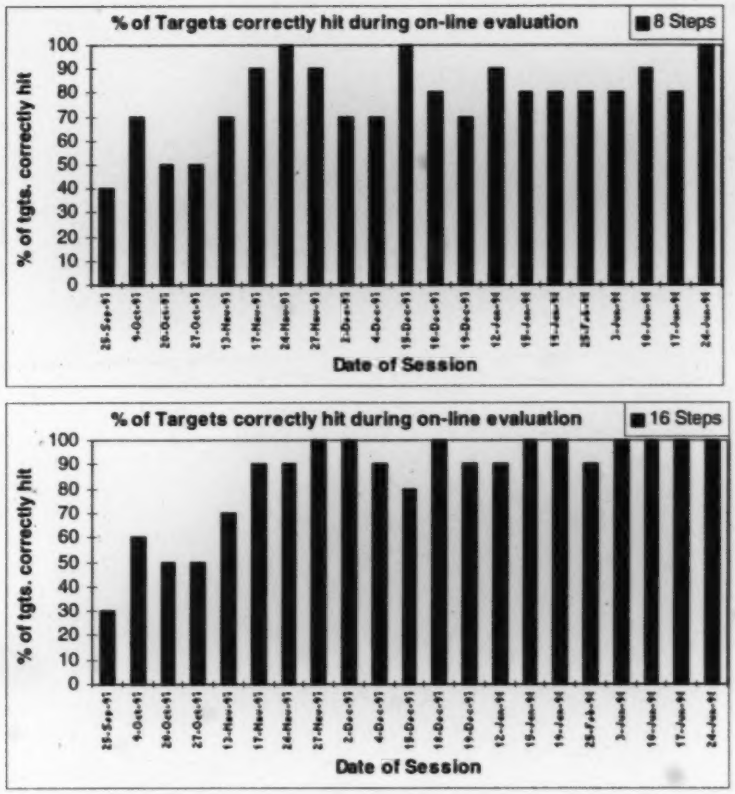

Figure 4-2. Accuracy achieved during on-line evaluation by subject L.G. Top figure shows the percentage of targets correctly hit during evaluation at a speed where a minimum of 8 cursor steps were required to hit or miss the target, while the bottom figure shows the results at a minimum of 16 steps.

An alternative to using specific frequency bands calculated by AR or FFT as features, is to use the AR model coefficients alone in a classification procedure [42]. The AR coefficients contain all the information of the signal that would be in the spectra; therefore, a classifier should be able to discriminate between sets of AR coefficients calculated for signals with different spectral properties. The exact frequency band that produces optimal results in a BCI system can vary according to the subject [2],[24],[45].

The advantage of classifying with raw AR coefficients is that one does not need to search for specific frequency components that contain the information.

The purpose of this section is to compare FFT and AR methods of feature extraction in BCI. We make no assumptions about where in the spectra the relevant information may be located and divide the spectra into evenly spaced power bands. We chose to compare FFT and AR spectral techniques because these are most common in BCI to date. We also compare these two methods to the method of using raw AR features for classification.

4.2 Methods

Experimental Setup

The subject's goal is to move the object on the screen to a target. The position of the target is switched between UP and DOWN at the end of each run when the object reaches the target or the opposite end of the screen is hit. We chose cursor movement because it is objectively measurable, easily implemented, simple for the user to learn, and can serve as a prototype for control of a wide variety of assistive devices.

Subject Training

We used three subjects in this study that acquired adequate control over the object on the screen; however, only one subject remained for longer than ten sessions. All our subjects were able to achieve some control before their fifth training session. Each of the sessions lasted approximately thirty minutes. The first half of each session was used to train a new classifier and the second half was used to evaluate the performance. Performance was evaluated in terms of how many times the target was hit versus missed at various movement speed of the object. During the sessions, the position of the object was updated every 50 milliseconds. In the training and evaluation of each subject, AR coefficients were used as features to the real-time classifier. The speed of the cursor was set by the number of full steps required to move from the center position to either the top or bottom edge of the screen. During training, at least 32 full steps were required to move the cursor to the target or miss. At 50 milliseconds per step this would be at least 1.6 seconds from the beginning of cursor movement, assuming that only full steps are taken in the direction of the target. To make the cursor take larger steps when the classifier is certain to which class (UP or

DOWN) the current signal pattern belongs as compared to cases of uncertainty, the distance of a full step is multiplied by the classifier confidence (0 to 1) in its output. Therefore usually only partial steps in either direction are taken by the cursor and the cursor does not move uninterrupted to the target, so the number of steps that was needed to move the cursor to the target was usually around two times the minimum. The evaluation performance of one of the subjects at minimum 8 and 16 steps to the target can be seen in Figure 4-2. At 8 steps to the target (0.8 seconds minimum from start of cursor movement) the results show greater variance than at 16 steps. These results are expected because at faster cursor movement, the subject has less chance to correct the cursor movement if it ever deviates from the correct direction. At 32 steps to the target this subject was able to get 100% accuracy in evaluation in every session after he first learned how to control the cursor movement (fifth session).

To ensure that artifacts, such as muscle or eye movement, were not affecting the cursor movement, we took the following steps:

- Watching, on a computer monitor, the real-time EEG throughout the experiment
- Closely watching and sometimes videotaping the subject throughout the experiment
- Examining the averaged EEG spectra after the experiment, especially during the initial sessions.

Off-Line Analysis

Figure 4-3 shows a flowchart of the three feature extraction processes compared in this study. A half-second window was used with a 0.45 second overlap to create the feature vectors. For the spectral methods, each window of signal was passed through a Hanning

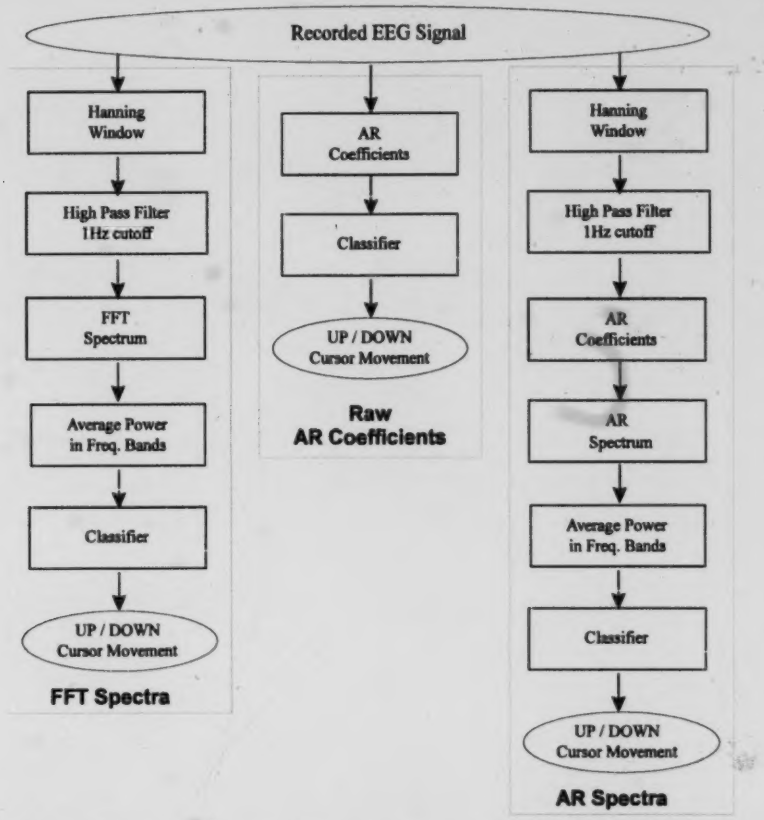

Figure 4-3. The feature extraction and classification process used in off-line analysis.

window and a high-pass filter with a 1 Hz cutoff to remove low frequency components. To get a discrete number of features from the spectrum, power values were averaged in equally spaced frequency bands. The spectral features were equally divided between 2 and 30 Hz. For example, taking four FFT features per lead gives, as input to the classifier, the average power in 2 to 9 Hz, 9 to 16 Hz, 16 to 23 Hz, and 23 to 30 Hz frequency bands. We used up

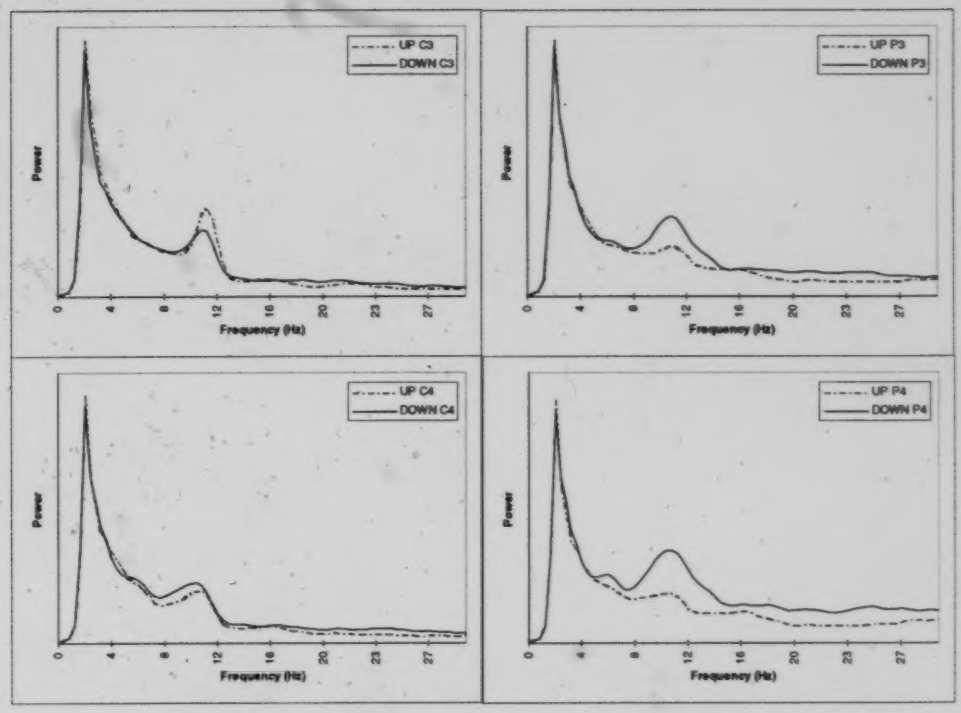

Figure 4-4. Average FFT modeled spectrum for the evaluation data in a sample session.

to 32 equally spaced frequency bands between 2 and 30 Hz giving a resolution of up to 0.875 Hz. Using four leads at this resolution, the training vector consists of 128 features; however, the high dimensionality may not be a problem because each training session contains approximately 5000 vectors. The AR model order was taken to be the same as the number of power bands. For example, when the average power in four frequency bands was used as the feature vector, a fourth order AR model was employed. Similarly, a 32nd order AR model was used when 32 frequency bands were selected. Using this method of choosing the order will give higher order and thus spectral detail when more frequency bands are needed, and similarly it will give lower order and detail for cases where the power is averaged in wide frequency bands. No prior knowledge was assumed of what frequency components may be most helpful with which subject in order to test the effectiveness of these feature extraction methods with new subjects. We extracted features for selected electrodes above the sensory-motor and parietal areas in both hemispheres (C3, C4, P3 and P4).

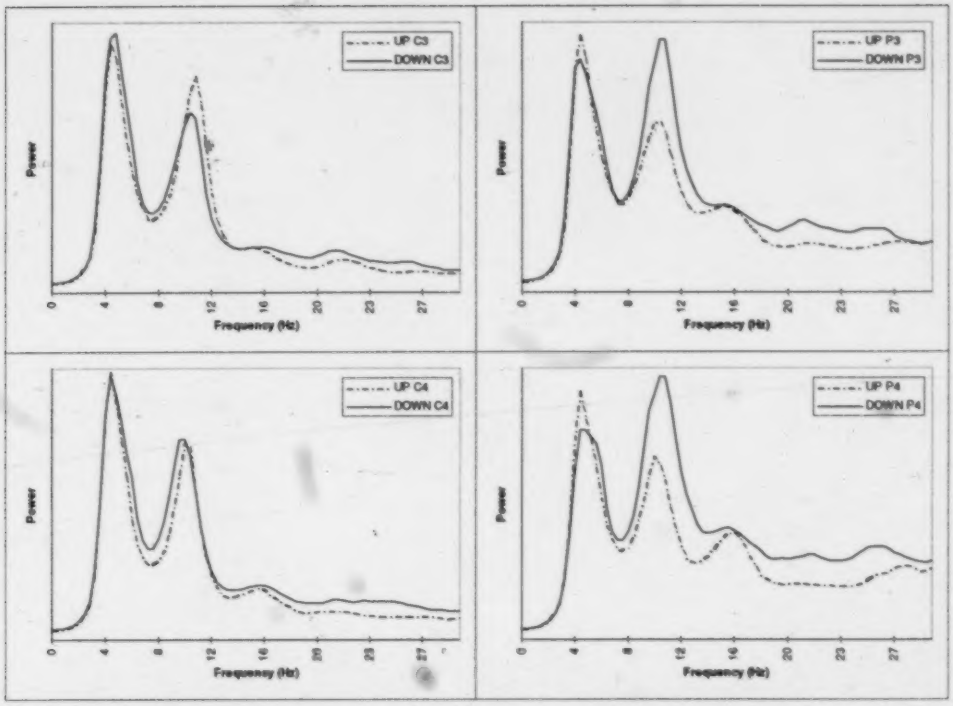

Figure 4-5. AR (32nd order) modeled spectrum for the coefficients whose average values can be seen in Figure 4-6.

The FFT algorithm that was used is from LabWindows CVI (National Instruments) and the AR algorithm was taken with minor modifications from Press et al. [43]. The correctness of these routines was cross-verified by the S-PLUS statistical software. The most common procedures for AR modeling are: Kalman filtering, the Yule-Walker approach, and the Burg algorithm. We selected to use the Burg algorithm because it was shown by Jansen [42] to give superior results in EEG analysis over the other two AR methods. An example plot showing the averaged FFT spectrum of the data used during a BCI session can be seen in Figure 4-4. This figure shows a significant difference between the up and down thought patterns in the μ-rhythm. The averaged AR coefficients can be seen in Figure 4-6. The averaged spectrum calculated from these AR coefficients can be seen in Figure 4-5.

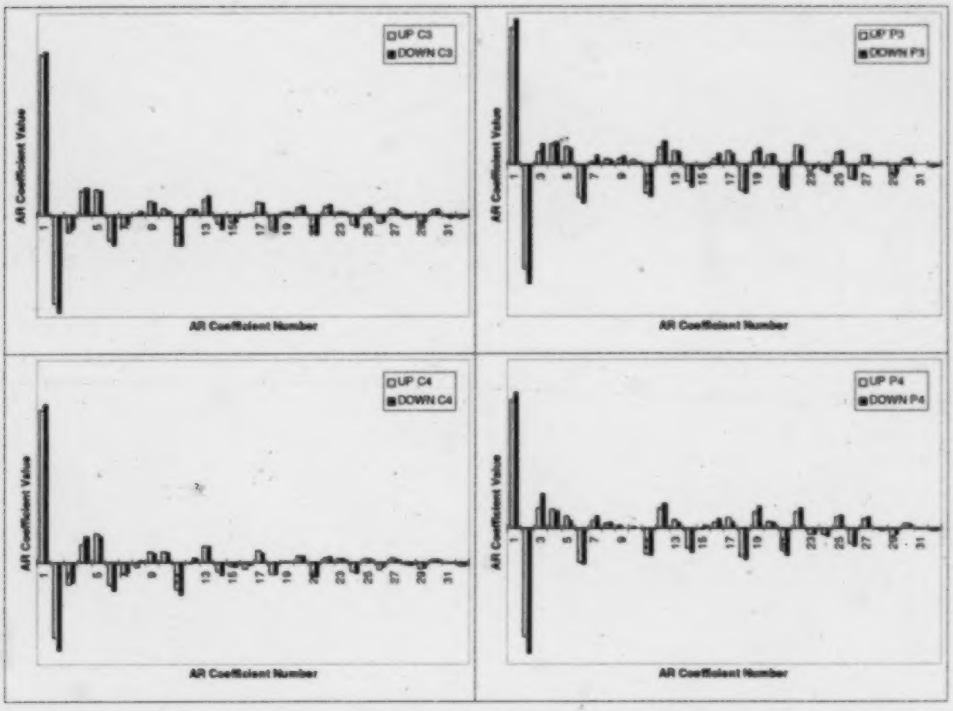

Figure 4-6. Average values of AR coefficients for the evaluation data of one session.

The quantitative measure of performance that we used is the probability of misclassification (PMC). PMC tells us the probability that each sample will be incorrectly classified, and is a common and intuitive measure of performance for classification problems. ALN was the classifier that we used in both the on-line experiments and in this analysis. ALN is a non-linear adaptive system which is capable of learning any continuous function to any degree of accuracy on a compact set [46].

4.3 Results

Figure 4-7 shows the off-line analysis results for several consecutive sessions at different numbers of features, using three different feature extraction methods. The method that uses raw AR coefficients shows a relatively large range of PMC for each session; however, this is the case because high PMC is generated with two AR coefficients. Taking four AR

coefficients or more narrows the difference in results between the various numbers of features used for classification.

There were three important events during the course of the sessions. In subject L.G.'s fifth session, the subject first achieved control of the cursor on the screen and it can be seen in Figure 4-7 as a large drop in PMC. After the first twelve sessions, we changed some setup parameters that made subject training easier. It is interesting to note that the off-line analysis results became worse once this improvement in on-line training was implemented. A possible explanation for the increase in off-line error is that when the on-line cursor control becomes easier, the subject needs to exert less mental effort and as a result the patterns in the EEG data are not as clear. After 18 sessions, subject L.G. took a break for 3 months, which was an interruption in his BCI training schedule. This break is associated with an increase in PMC on the following session.

Figure 4-8 shows the length of time that was needed to train a new classifier during the on-line sessions. A line of regression shows the general decrease in the required training time, which suggests that the subject is becoming more adept at controlling the BCI.

To compare the amount of information that is captured by each feature extraction method which is relevant to discriminating between the UP and DOWN cursor movement patterns, we compared the results of the three methods taken with the optimal number of features for each method and session. The analysis results for three subjects can be seen in Figure 4-9. In this figure, "AR" represents raw AR coefficients being classified, "FFT" represents power bands approximated by FFT being used for classification, and "ARspec" stands for power bands approximated by AR. Figure 4-9 shows that classifying the correct number of raw AR coefficients gave better results, in the majority of the sessions, than using the other feature extraction methods. The averaged results of all the sessions for each subject can be seen in Figure 4-10, which shows that using AR coefficients as features for classification gave slightly better, or similar results to the other two methods.

For subject L.G., the difference between the feature extraction methods that gave the best results (raw AR coefficients), and the worst results (FFT) is less than two percent, so for this subject the three methods can be assumed to be equal in terms of classification power. Subject L.U. showed an approximately three percent difference in PMC between the best and worst methods, which again is insignificant. Only the data recorded for subject R.J. showed a significant eight percent difference.

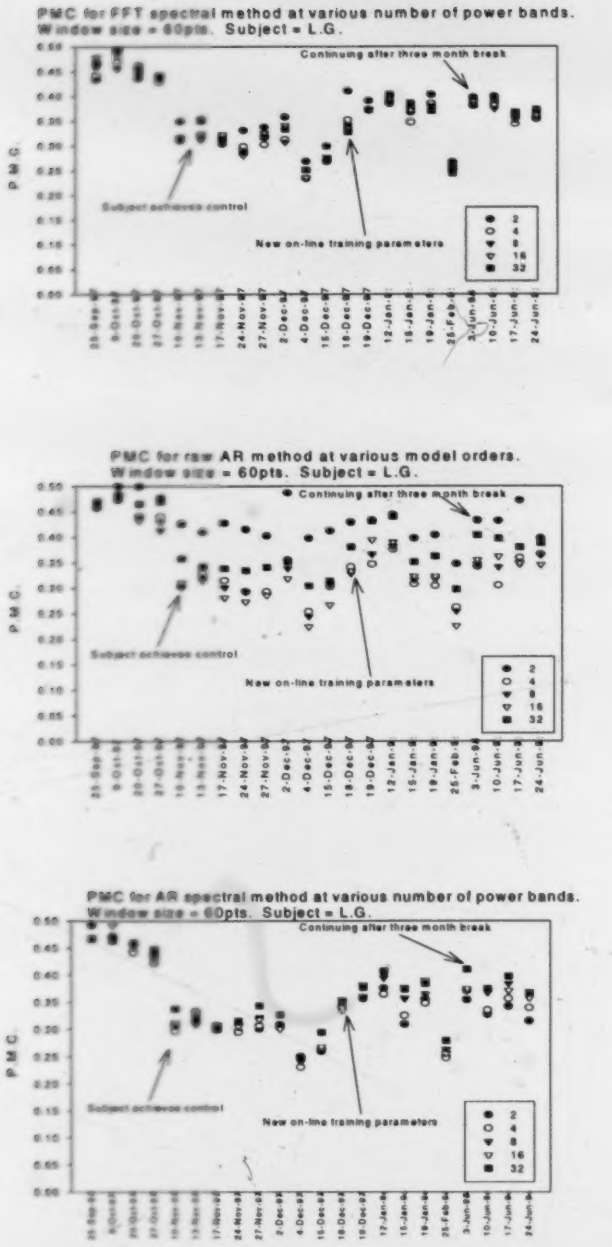

Figure 4-7. Off-line analysis for all the sessions with one BCI subject using the three methods of feature extraction at different numbers of features per lead.

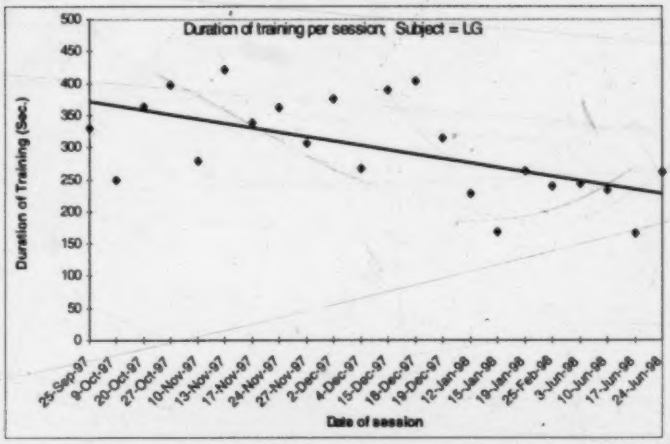

Figure 4-8. The duration of training per session for subject L.G.

The number of features that was required by each method can be determined from Figure 4-11. This figure shows that both the methods of feature extraction that used spectral bands needed only two features per lead. These two features were the average power in 2 to 16 and 16 to 30 Hz frequency bands. Since the μ rhythm (8-12 Hz) is the classical feature used for discrimination in BCI systems that employ voluntary modulated EEG, and is included in the 2 to 16 Hz frequency band, this may explain why only two frequency bands were needed by the spectral methods. It can be seen from Figure 4-11 that for all three subjects the error rate decreases only slightly, if at all, by subdividing the 2 to 16 and 16 to 32 Hz frequency bands into more narrow bands. The raw AR coefficients method, however, showed a significant drop in error between two and four features per lead in all three subjects. Figure 4-11 suggests that adding more than four AR coefficients per lead does not provide much more useful information to the classifier because doing so did not significantly improve classification results. Although four AR coefficients per lead gave the best results with R.J. and almost the best results with the other subjects, L.G. had the lowest PMC with sixteen AR coefficients, and subject L.U. with eight coefficients, but the difference compared to four AR coefficients not significant (less than 2% in each case).

Figure 4-11 plots the off-line analysis results done with a window size of 60 points (sampled at 200 Hz), and a window size of 100 points. The reason for comparing the results at different window sizes is that a large number of features can be approximated to greater accuracy with more samples, and also because some feature extraction methods may

be more affected by short window size than other. As Figure 4-11 shows, the PMC did decrease with all three subjects when the window size was increased to 100 points, but the decrease in PMC was similar for all feature extraction methods and numbers of features.

4.4 Conclusions for Feature Extraction

With a BCI that utilizes parallel man-machine training, the human subject and computer need to learn at the same time. With a new subject, it is not known exactly which frequency components are going to be most useful in discriminating among the EEG patterns. Without knowing which frequencies contain useful information, raw AR coefficients give equal or slightly better classification results and require less processing compared to FFT and AR spectral approximation methods.

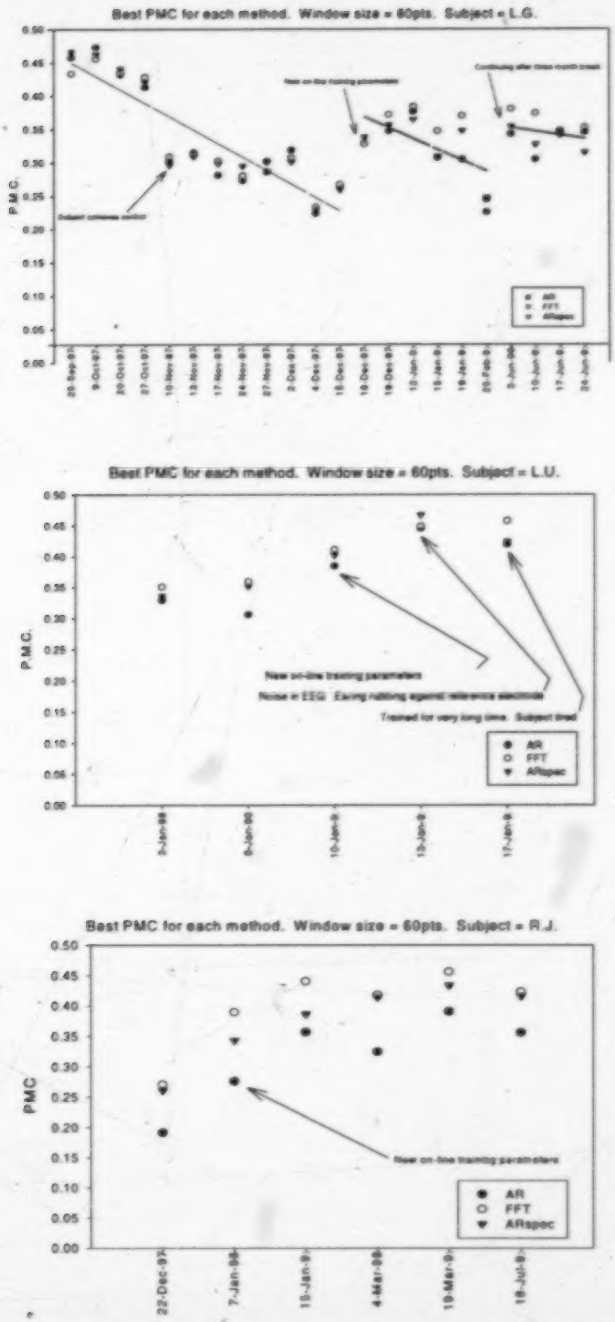

Figure 4-9. Off-line analysis for all the sessions with three BCI subjects using the three methods of feature extraction at the best number of features.

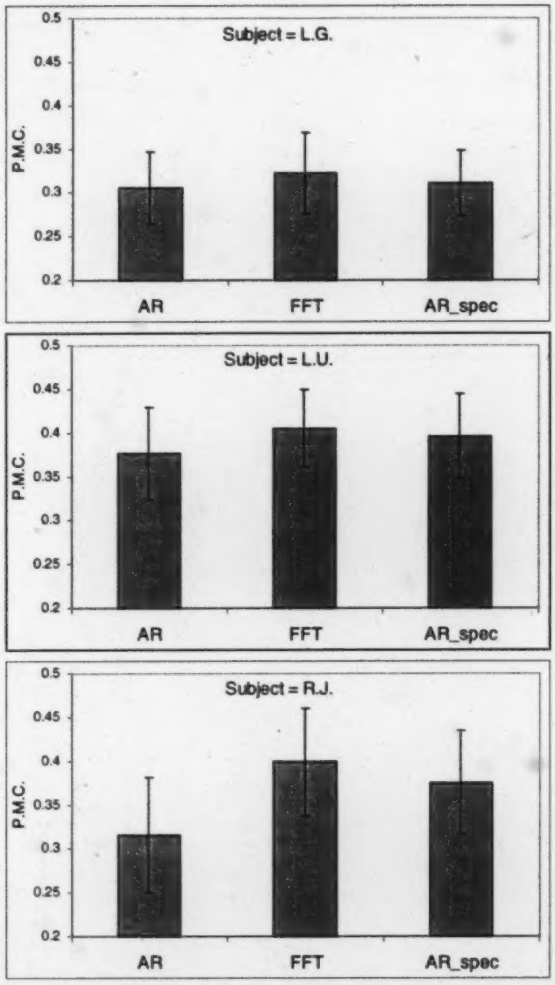

Figure 4-10. A comparison of the three feature extraction methods by three subjects. The best number of features was taken for each method in each session. All sessions, after cursor control was achieved, were averaged at a window size of 60 samples. The error bars show the standard deviation in PMC between the sessions of each subject.

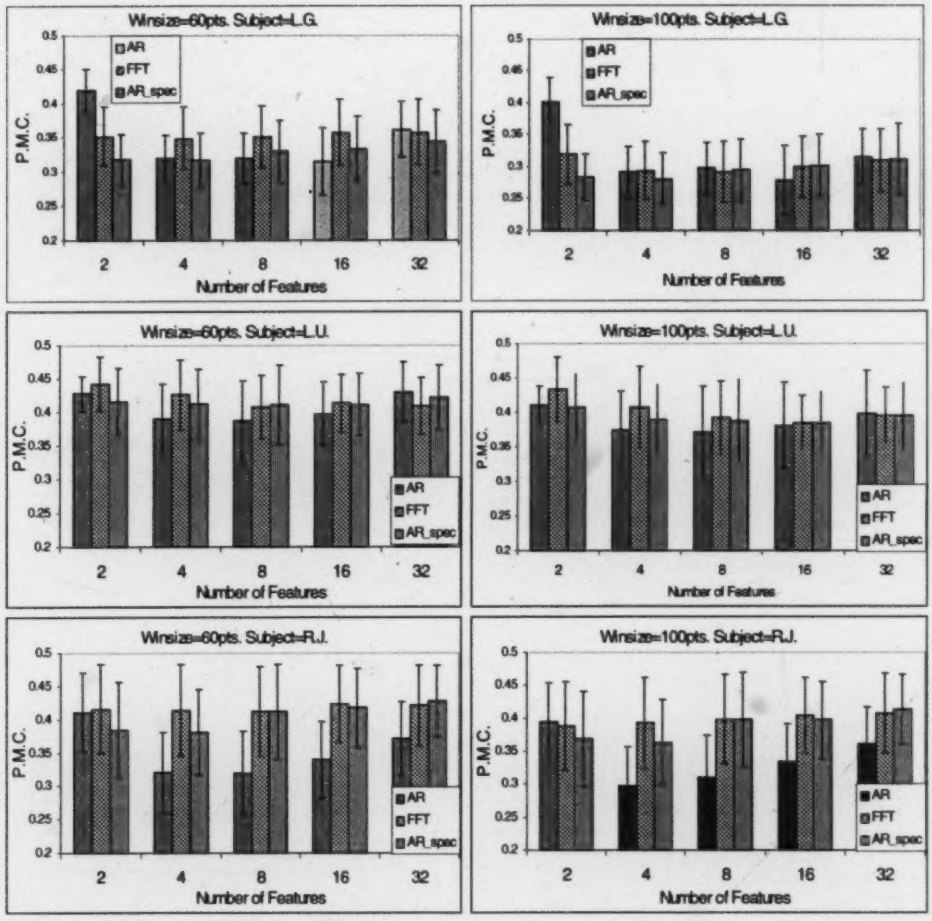

Figure 4-11. Average PMC in off-line analysis of the three feature extraction methods. All sessions, after cursor control was achieved, were averaged separately for three subjects. The left figures show the results with a window size of 60 points (at 200 Hz), while the right figures show the results with a window size of 100 points. The error bars show the standard deviation in PMC between the sessions of each subject.

Chapter 5

Classification

The problem of classification is that of partitioning the feature space into regions, one region for each category, according to labeled training data. As one would expect, the problem of unsupervised learning, where no labels are given, is a more difficult one. The category to which each EEG sample belongs (UP, DOWN, LEFT, RIGHT) is always known in our application, so we are doing supervised learning.

In this section, different classifiers are evaluated on off-line data. The data was recorded from a real-time experiment in which a subject was controlling a cursor on a screen in the UP and DOWN direction. The objective is to assess the pattern recognition component in our system by comparing the classification performance of the presently used ALN with:

1. K-nearest neighbour classifier (K-NN)
2. Classification trees
3. Quadratic discriminant and linear discriminant.
4. Multilayer perceptron with the back-propagation training law (MLP-BP)
5. Three versions of the Learning Vector Quantization (LVQ) algorithm

No single classifier is optimal for every type of problem. The performance of a classifier depends on the nature of the data being tested such as dimensionality, number of training samples, and class distribution (ex. Normal, bimodal,...) To date, an extensive comparison of classifiers has not been done for BCI problems.

5.1 Experimental Setup

Randomness in Adaptive Classifiers

The pattern recognition models, such as MLP-BP, LVQ, and ALN, which have random initial parameters, gave slightly different results each time they were executed on the same data. To minimize the variance in results caused by the random nature of these classifiers, the classifiers were trained and evaluated five times and the median value was taken. The

median was used instead of the mean, because a calculated mean result would not have been actually achieved by any of the classification runs and instead it would be artificially approximated.

Training and validation sets

We trained and tested all classifiers on data recorded for two subjects, L.G. and F.K. The data recorded during the training phase of the experiment was used for building the classification model, which includes optimizing the classifier parameters (if needed) and training. The data recorded during the testing phase of the experiment was set aside and used for testing the performance of each classifier after the classification model was determined. The test set was in no way used for optimizing any classifier parameters. For adaptive models that need to have some parameters adjusted, the training set was split into

Subject = L.G.	DOWN	UP	Total
Train Set	2631	3249	5880
Testing Set	5401	3765	9166

Table 5-1. Number of samples of each class in the training and testing sets for subject L.G.

Subject = F.K.	DOWN	UP	Total
Train Set	5400	3160	8560
Testing Set	232	281	513

Table 5-2. Number of samples of each class in the training and testing sets for subject F.K.

two subsets, training set and validation set. These parameters need to be adjusted according to the characteristics of the data and usually address the stability-plasticity dilemma. The network is supposed to be plastic enough to learn an important pattern. But at the same time

it should remain stable when, in short-term memory, it encounters some distorted versions of the same patterns.

The training set for subject L.G. contained 2631 DOWN samples and 3249 UP samples, while the testing set had 5401 DOWN and 3765 UP samples as can be seen in Table 5-1. When needed by the classifier to experimentally approximate some training parameters, the training set was further divided into a validation set and a training set. Once the classifier parameters were determined, the classifier was retrained with the complete training set and tested on the testing set. The training and testing set for subject F.K. can be seen in Table 5-2. Subject F.K. took longer time during training than subject L.G. and became fatigued earlier during the evaluation, so the number of evaluation vectors for subject F.K. is small relative to L.G.

Pre-recorded data was used to generate the training and testing vectors. Autoregressive coefficients were calculated from a 0.3 second overlapping window for four leads and were used as inputs to the classifier to discriminate between the UP and DOWN classes. Two data sets were tested. One data set had four AR coefficients for each of four leads for a total of 16 input features. The second data set had eight AR coefficients for each of the four leads for a total of 32 input features. As was shown in the feature extraction section, four AR coefficients were optimal or nearly optimal for the majority of the cases. Eight AR coefficients per lead were also used to see how the classifier would handle cases of high dimensionality (32 inputs) where some variables may be redundant.

5.2 Classifiers

Adaptive Logic Network

Description

The ALN development system, which has been applied to problems in diverse areas including cardiology [48], predictive maintenance [49], rehabilitation [50], non-destructive testing and high-energy physics, was provided by Dendronic Decisions Limited [46]. The ALN is an artificial neural network that can approximate any continuous, real-valued function on N-dimensional space. The difference between the ALN and other neural networks is that the ALN utilizes piecewise linear functions to approximate the function being modeled. The ALN has a number of linear functions acting on the network inputs, and their outputs feed into a tree of MAX (maximum) and MIN (minimum) operators. In this way, an ALN can approximate any continuous function defined on a compact set to any degree of precision. Direct application of MAX to linear functions produces a convex-down function (surface like a bowl) while direct application of MIN to linear functions produces a convex-up function (surface like a dome). The MIN and MAX operators that take inputs from linear functions make up the ALN logic tree. The function graph is generally composed of parts of different curvatures, which can efficiently be fitted by an appropriate ALN tree of MAX and MIN operators.

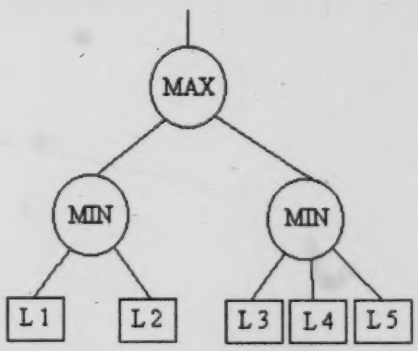

Figure 5-1. Structure of MAX and MIN operators, and linear pieces (L1, ... L5), which could be used to generate the function in Figure 5-2.

A typical ALN structure of MAX and MIN operators and linear functions can be seen in Figure 5-1 and an example of the function that this structure can create is shown in Figure 5-2.

During training, the function graph or surface is modified to fit the points in the training set. This is done by an algorithm that, given an input vector and its desired output, modifies the weights of the appropriate linear function. A linear piece (a part of a hyperplane) moves and tilts to get closer to training points near it. The linear functions in an ALN are responsible only for fitting points on them or directly above or below them, so that only a single linear function is made responsible for each data point. The fact that only one linear piece is made responsible for each data point is an important difference between adaptive logic networks and backpropagation neural networks, where the use of sigmoids results in distribution of responsibility for error over all inputs. If there were only one linear piece in the ALN, the training procedure would be just a special case of linear regression using the input variable and a constant as basis function.

The following advantages are the main reasons for using ALN in the brain-computer interface:

- Speed of training: In order to do on-line training at each step, a very fast algorithm is required. This requirement excludes most classifiers. With ALN only one linear function is made responsible for each data point so only one linear element needs to be adjusted at

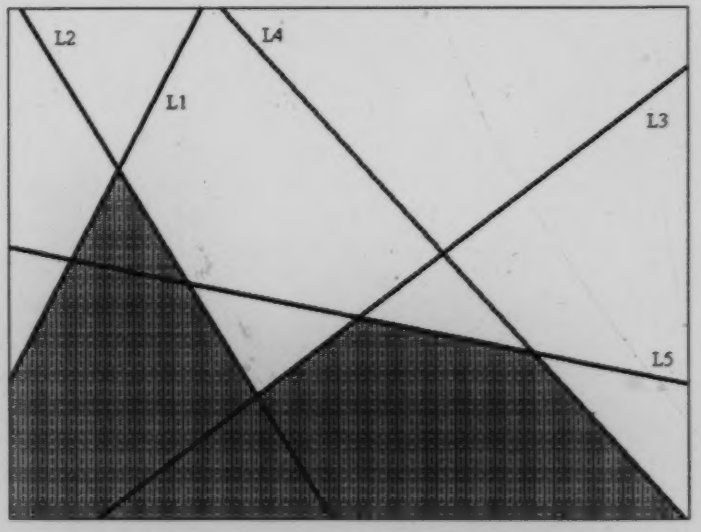

Figure 5-2. A simple example of an ALN approximated function.

each training step.

- Speed of evaluation: To meet the timing constraints during on-line training, the classifier must be able to evaluate as well as train quickly. ALN is very fast in evaluation. Some classifiers such as probabilistic neural networks have unacceptably slow evaluation and therefore could not be used in our system.
- Information on importance of each input: In most cases increasing the number of features will cause the performance to improve. At worst, the classifier should ignore the new features, and if the new features provide any additional information, the performance should improve. Unfortunately in practice the inclusion of additional features beyond a certain point will lead to worse rather than better performance. The basic source of the problem can be traced to the fact that the number of training samples is finite. Therefore it is important to be able to see the contribution that each input has to the output of the classifier so that an input parameter can be removed if its contribution is insignificant. With ALN it is possible to see the significance of each input feature by looking at the respective linear function coefficients (weights) in the trained tree. Although this is an important feature of ALN, it has not been used in this project.
- Control of generalization: Overtraining is a problem in most non-linear classifiers. If a neural network is trained too long it will learn the training data too well and lose the ability to generalize to a test set drawn from the same source as the training data. ALN overcomes this problem by letting the user limit the number of linear pieces or by imposing bounds on the weights of the linear pieces
- The size of the ALN tree does not need to be static. The depth of the ALN tree can be manually set, or the tree can be automatically built by the ALN. If the tree is to be automatically built, then training starts with one linear piece, and the ALN splits linear pieces if, after adjustment of weights, the RMS error is greater than an acceptable error level on all pieces. In this way, new linear pieces keeps getting added until an acceptable error level is reached. The ability of ALN to grow its structure can be a useful property in BCI training. With ALN, assumptions do not have to be made about the complexity of the control function, instead it can be determined during the real-time experiment

Error Tolerance

The ALN has a tolerance parameter that needs to be adjusted for each input feature. Tolerance is a measure of the allowable error (or smoothing) for each variable. A high value of tolerance tends to reduce the magnitude of changes in weight for a given variable during training. A low value of tolerance may allow such large changes of weight that a linear piece could tilt back and forth in rapid oscillations trying to fit data points. In cases like this where one does not have knowledge of what the allowable error should be for each input, an automated method must be determined to approximate this parameter.

In a normalized input space where the range of each variable is from 0 to 1, let x[i] be a component of a single training sample. i is an integer from 1 to d, and d represents the dimensionality of the vector (number of variables). The influence of each training sample is from (x[i] – *tolerance*[i]) to (x[i] + *tolerance*[i]). The range of influence of a sample along each axis is therefore $2 \cdot tolerance[i]$.

Assuming a hypercube for volumetric considerations, the total volume of the input space is 1. Also assuming equally distributed samples, then in order to cover the entire input space with N samples the hypercube around each vector needs to cover a volume equal to $1/N$; therefore, the sample, needs to cover a length of $(1/N)^{1/d}$ in each dimension. Equating this length with the range of influence for each variable gives the following equation:

$$2 \cdot tolerance[i] = \sqrt[d]{\frac{1}{N}}$$

$$tolerance[i] = \frac{1}{2}\sqrt[d]{\frac{1}{N}}$$

The above equation could be used to approximate the *tolerance* if all the variables in the sample were in the range of 0 to 1. If the samples are not normalized, then *tolerance* in this equation needs to be multiplied by the range of each variable $\Re[i]$, giving the following equation.

$$tolerance[i] = \frac{\Re[i]}{2} \cdot \sqrt[d]{\frac{1}{N}}$$

In most real-world problems, the acquired samples are not uniformly distributed across a defined range of the input space. In many cases, the possible range may be unknown or even infinite. A more common distribution is the normal distribution where mean plus and minus two times the standard deviation covers 95% of the samples. To cover 95% of the samples therefore requires the range of four standard deviations. If we replace the range in the *tolerance* equation by four standard deviations we get a more practical approximation of:

$$tolerance[i] = 2 \cdot stdev(i) \cdot \sqrt[4]{\frac{1}{N}}$$

where *stdev(i)* is the standard deviation of i^{th} feature. However, this assumes that one data point is enough to determine the function values. If there is a lot of noise, the points will have to cover more territory and *tolerance* will have to be larger. If we want to have a factor of K times coverage, then we replace $1/N$ in the above equation by K/N. Effectively we are saying that it will take K times the minimal number of points to cover the same volume of space. The new equation for *tolerance* can be written as:

$$tolerance[i] = 2 \cdot stdev(i) \cdot \sqrt[4]{\frac{K}{N}}$$

Although a better approximation may be to have tolerances that vary according to the position in the input space, the above method is an effective and computationally fast approximation.

Using the above equation, the optimal value of K was approximated by varying K while observing the probability of misclassification (PMC) on the reserved data from the training set. This way the ALN was trained and evaluated on separate data. Figure 5-3 and Figure 5-4 show a plot of PMC on the reserved data from the training set with respect to different values of K, for subject L.G. As can be seen from these figures, there is little difference in

expected results for values of K between 2 and 4. In the 16 feature example, K having the value of 5 still gave similar results, but in the 32 feature example, the error increased because of the higher dimensionality since each point now has to cover greater volume. In general for both the 16 and 32 feature data set, the best results seem to be at $K = 2$.

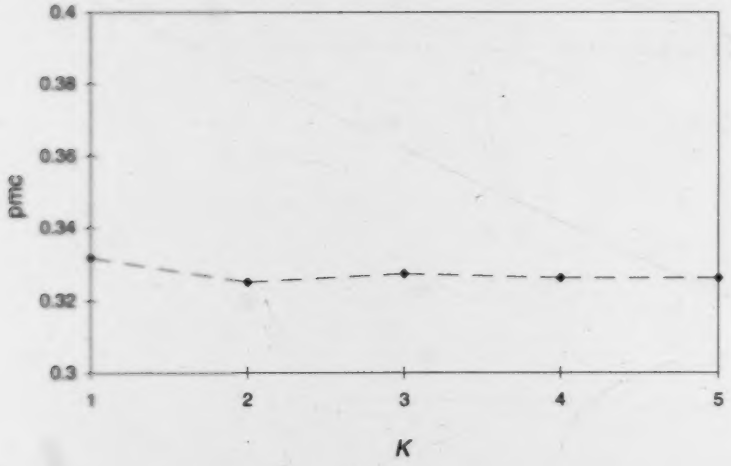

Figure 5-3. Value of K used in calculating tolerances vs. PMC for subject L.G. at 16 input features.

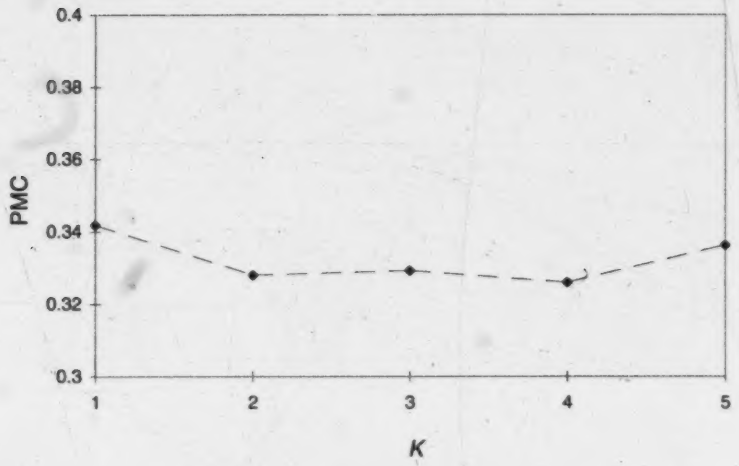

Figure 5-4. Value of *K* used in calculating tolerances vs. PMC for subject L.G. at 32 input features.

In subject F.K., the value of *K* with respect to PMC had similar characteristics with nearly equal results at 2, 3, and 4 for both 16 and 32 feature examples. Although *K* having the value of 4 approximated the best results, the difference in PMC between *K* being 2 and 4 was less than 0.01. In order to have classifier parameters that can generalize for both subjects the value of 2 was used for *K* in both subjects.

Results

A convenient method for displaying classifier results is a confusion matrix. Confusion matrix refers to the matrix of counts of the events "true class *i* decided as *j*". The columns of the confusion matrix indicate the number of occurrences that the classifier predicted DOWN

		Predicted	
		DOWN	UP
True	DOWN	3710	1691
	UP	1306	2459

Table 5-3. Confusion matrix for the median run of ALN with *K*=2 on the 16 feature data for subject L.G.

		Predicted	
		DOWN	UP
True	DOWN	3690	1711
	UP	1315	2450

Table 5-4. Confusion matrix for the median run of ALN with *K*=2 on the 32 feature data for subject L.G.

and UP classes, while the rows of the matrix indicate what the true class was for those occurrences. Using *K*=2, the network was retrained with all the training data and tested on the evaluation set. The ALN was trained and evaluated five times. The median PMC for the 16 feature data set was 0.3300. The confusion matrix of the median run on the 16 feature test data is given in Table 5-3. The 32 feature data set produced the results in Table 5-4, which gives a PMC of 0.3330.

The classification results for subject F.K. can be seen in Tables 5-5 and 5-6 that give a PMC of 0.1052 and 0.1350 respectively.

		Predicted	
		DOWN	UP
True	DOWN	222	10
	UP	47	234

Table 5-5. Confusion matrix for the median run of ALN with *K*=2 on the 16 feature data for subject F.K.

		Predicted	
		DOWN	UP
True	DOWN	228	4
	UP	71	210

Table 5-6. Confusion matrix for the median run of ALN with *K*=2 on the 32 feature data for subject F.K.

K-Nearest Neighbor

Description

Let X_i be a sample in the testing set, and let X_j' be a sample in the training set nearest to X_i. Then the nearest-neighbor rule is to assign X_i the class associated with X_j'. An obvious extension of the nearest-neighbor rule is the k-nearest-neighbor (K-NN) rule. This rule classifies X_i by assigning it the class most frequently represented among the k nearest samples. In other words, classification is made by examining the classes of the k nearest neighbors of X_i and taking a vote.

K-Approximation

The K-NN algorithm was tried with 1, 2, 4, 8, 16, 32, 64, 128, and 256 nearest neighbors.

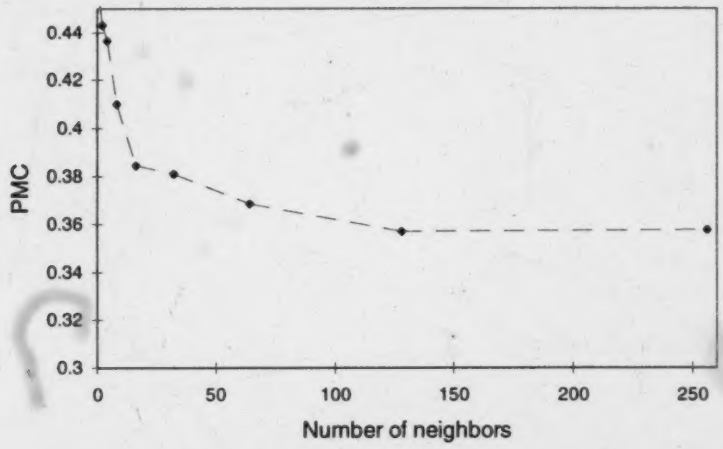

Figure 5-5. Number of neighbors vs. PMC for the K-NN method with 16 feature data. Subject = L.G.

For subject L.G., the optimal number of neighbors was 128 in both the 16 and 32 feature data set, as can be seen in Figure 5-5 and Figure 5-6. For subject F.K., the best results were predicted with a 64 neighbors in the 16 feature data and 128 in the 32 feature data; however, the difference in PMC was negligible so 128 neighbors were taken in all cases. The plot of PMC versus number of neighbors for F.K. can be seen in Figure 5-7 and Figure 5-8.

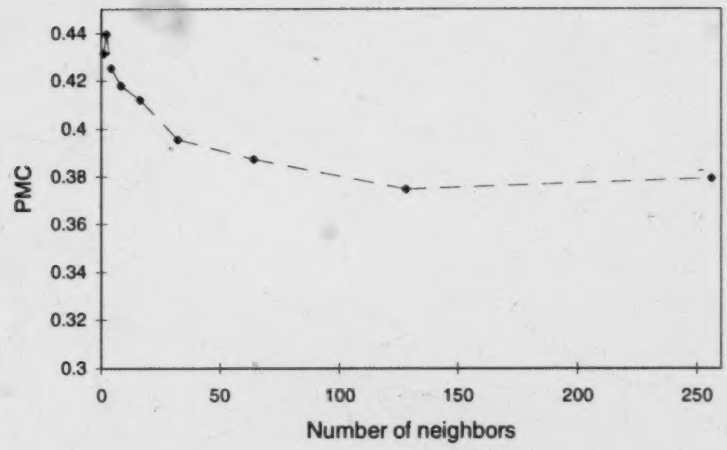

Figure 5-6. Number of neighbors vs. PMC for the K-NN method with 32 feature data. Subject = L.G.

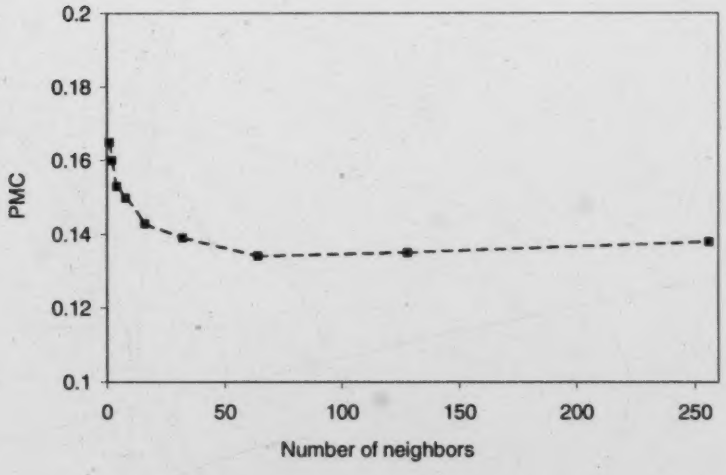

Figure 5-7. Number of neighbors vs. PMC for the K-NN method with 16 feature data. Subject = F.K.

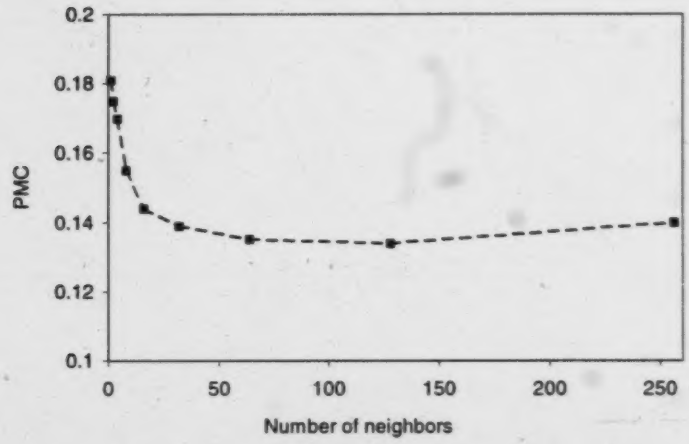

Figure 5-8. Number of neighbors vs. PMC for the K-NN method with 32 feature data. Subject = F.K.

		Predicted	
		DOWN	UP
True	DOWN	2457	2944
	UP	769	2996

Table 5-7. Confusion matrix for 128-NN on the 32 feature evaluation data with subject L.G.

		Predicted	
		DOWN	UP
True	DOWN	2511	2890
	UP	838	2927

Table 5-8. Confusion matrix for 128-NN on the 16 feature evaluation data with subject L.G.

Taking K=128, and classifying the data for subject L.G. according to the full training set, the PMC in the evaluation set was 0.3788 for the 16 feature data set and 0.3747 for the 32 feature data set. The corresponding confusion matrices can be seen in Table 5-8 and Table 5-7.

For subject F.K., taking K=128, and evaluating with the full train set, the PMC in the evaluation set was 0.1528 for the 16 feature data set and 0..1522 for the 32 feature data set. The corresponding confusion matrices can be seen in Table 5-9 and Table 5-10. It should be noted that for both subjects, the results were very similar in the 16 and 32 feature data.

		Predicted	
		DOWN	UP
True	DOWN	223	9
	UP	75	206

Table 5-9. Confusion matrix for 128-NN on the 16 feature evaluation data with subject F.K.

		Predicted	
		DOWN	UP
True	DOWN	220	12
	UP	71	210

Table 5-10. Confusion matrix for 128-NN on the 32 feature evaluation data with subject F.K.

Classification Tree

Description

The use of classification trees is most widely known in applications of botany and medical diagnosis because this classifier is easy to comprehend and thus have confidence in [65]. Samples are passed down the tree, starting at the top node, with decisions being made at each node until a terminal node or leaf is reached. Each non-terminal node contains a question on which a split is based. Each leaf contains the label of a classification. A classification tree partitions the input space into sub-regions corresponding to the leaves, since each sample will be classified by the label of the leaf it reaches.

The idea of tree induction is to construct a decision tree from a set of examples. Constructing a decision tree is usually done by growing the tree, that is by successively splitting leaves. Tree construction is easiest when there is an exact partition of the input space and every example can be classified correctly. In a noisy classification problem, where the samples of different classes overlap, growing the tree until every sample is classified correctly would over-fit the training set. A possible strategy is to stop growing the tree early, or another strategy is to prune the tree after constructing it. The main difference between algorithms for tree construction are the pruning strategy and the exact rule for splitting nodes. The model tested here was fitted using binary recursive partitioning whereby the data are successively split along coordinate axes of the variables so that at any node, the split which maximally distinguishes the response variable in the left and right branches is selected. Splitting continues until nodes are pure or data are too sparse. Classification trees are discussed in detail in [51].

Results

With the 16 feature data set, the PMC for subject L.G. was 0.3669 and the confusion matrix is given in. With the 32 feature data set, the PMC was 0.4247 and the confusion matrix is given in Table 5-13 and Table 5-12.

		Predicted	
		DOWN	UP
True	DOWN	220	12
	UP	75	206

Table 5-11. Confusion matrix for classification tree on the 16 feature evaluation data for subject F.K.

		Predicted	
		DOWN	UP
True	DOWN	2937	1463
	UP	2464	2302

Table 5-12. Confusion matrix for classification tree on the 32 feature evaluation data for subject L.G.

		Predicted	
		DOWN	UP
True	DOWN	3567	1834
	UP	1484	2281

Table 5-13. Confusion matrix for classification tree on the 16 feature evaluation data for subject L.G.

		Predicted	
		DOWN	UP
True	DOWN	217	15
	UP	78	203

Table 5-14. Confusion matrix for classification tree on the 32 feature evaluation data for subject F.K.

With the 16 feature data set, the PMC for subject F.K. was 0.1593 and the confusion matrix is given in Table 5-11. With the 32 feature data set, the PMC was 0.1711 and the confusion matrix is given in
Table 5-14.

Linear and Quadratic Discriminant Analysis

Description

Linear discriminant analysis assumes that a linear barrier or a hyperplane exists such that the set of inputs that compose one class lie on one side of the barrier while the other points on the other side. Figure 5-9 shows distributions for two classes being separated by a linear discriminant. Fisher's (1936) linear discriminant implementation was used, which was first described in [52]. Quadratic discriminant analysis assumes that a curved barrier exists that can separate the two classes. The quadratic rule goes back to C.A.B. Smith (1947) [53]. Even though the quadratic method is guaranteed to outperform the linear rule for very large sample sizes, it can be outperformed by the linear rule for moderate sample sizes.

Results

For subject L.G., on the 16 feature data set, linear discriminant analysis gave PMC of 0.3175 and a quadratic discriminant gave a PMC of 0.3776, while on the 32 feature data set, a linear discriminant gave PMC of 0. 3235 and a quadratic discriminant gave PMC of 0. 3878. The confusion matrices can be found in Table 5-15, Table 5-16, Table 5-17, and Table 5-18.

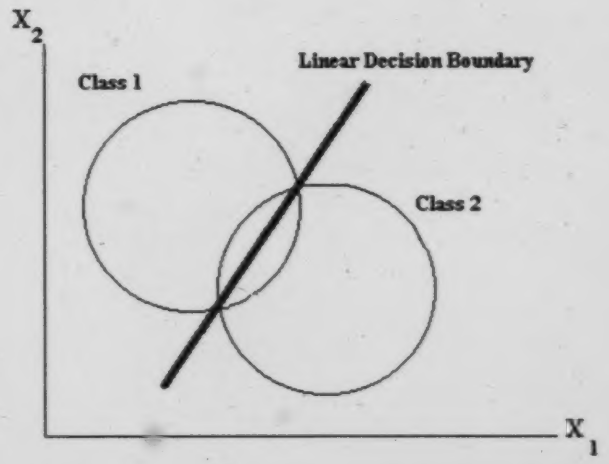

Figure 5-9. Example of a two class linear discriminant.

		Predicted	
		DOWN	UP
True	DOWN	3993	1408
	UP	1409	2356

Table 5-15. Confusion matrix for LDA on the 16 feature data for subject L.G.

		Predicted	
		DOWN	UP
True	DOWN	4277	1124
	UP	2060	1705

Table 5-16. Confusion matrix for QDA on the 16 feature data for subject L.G.

		Predicted	
		DOWN	UP
True	DOWN	3492	1909
	UP	1105	2660

Table 5-17. Confusion matrix for LDA on the 32 feature data for subject L.G.

		Predicted	
		DOWN	UP
True	DOWN	3938	1463
	UP	1900	1865

Table 5-18. Confusion matrix for QDA on the 32 feature data for subject L.G.

For subject F.K., on the 16 feature data set, linear discriminant gave a PMC of 0.1258 and quadratic discriminant gave PMC of 0.1201, while on the 32 feature data set, linear discriminant analysis gave PMC of 0.1332 and quadratic discriminant gave PMC of 0.1453. The confusion matrices can be found in Table 5-19, Table 5-20, Table 5-21, and Table 5-22.

		Predicted	
		DOWN	UP
True	DOWN	224	8
	UP	61	220

Table 5-19. Confusion matrix for LDA on the 16 feature data for subject F.K.

		Predicted	
		DOWN	UP
True	DOWN	225	7
	UP	59	222

Table 5-20. Confusion matrix for QDA on the 16 feature data for subject F.K.

		Predicted	
		DOWN	UP
True	DOWN	228	4
	UP	70	211

Table 5-21. Confusion matrix for LDA on the 32 feature data for subject F.K.

		Predicted	
		DOWN	UP
True	DOWN	226	6
	UP	71	210

Table 5-22. Confusion matrix for QDA on the 32 feature data for subject F.K.

It should be noted that with 16 input features, quadratic discriminant gave slightly better results for subject F.K., while linear discriminant gave better result for subject L.G. This seems to suggest that the nature of the classification function varies between subjects. For 32 input features, both subjects had better results with linear discriminant. This result is expected since linear discriminant usually gives better result in cases of high dimensionality and small or moderate data sets.

Multi-Layer Perceptron with Back Propagation learning law

Description

Multi-layer perceptron neural networks are composed of many simple elements called neurons, interconnected in a parallel architecture. Each connection coming to the neuron is called a synapse and has an associated weight W0 - Wn (Figure 5-10). A neuron has one output, which is generally related to the state of the neuron, and may fan out to several other neurons. The inputs are the activations of the incoming neurons multiplied by the weights of the synapses. The activation of the neuron is computed by applying a "squashing" function to this product. This squashing function is generally some form of nonlinear function. One popular class of functions used is the sigmoid, or logistic function.

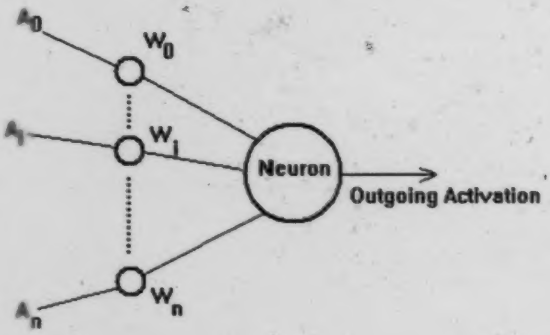

Figure 5-10. Diagram of abstract neuron model.

A neural network is usually made up of several layers of neurons. The first layer of neurons is called the input layer. This layer receives an input vector and fans it out onto the next layer. The final neurons coming out of the network make up the output layer. The values appearing at the output neurons are the results of the calculation. The layers between the input layer and the output layer are called hidden layers. An example of a neural network with 3 input neurons, 1 hidden layer of 2 neurons, and 1 output neuron can be seen in Figure 5-11.

The backpropagation algorithm was developed by Paul Werbos and is the most common in use. It is estimated that over 80% of all neural network projects in development use backpropagation [54]. In backpropagation, there are two phases in its learning cycle, one to propagate the input pattern and the other to adapt the output. Each time an input vector is presented to the network, a result is calculated according to the weights of the synapses. If the calculated result matches the correct output of the input vector, no learning takes place, and the next data sample is considered. However, if the calculated result appearing at the output neuron is wrong, the responsibility of this error is distributed among the lower level neurons, according to the magnitude of the influence on the output coming from them. The error signals are backpropagated in the network to the hidden layers according to the chain rule for differentiation. The responsible neurons have their weights adjusted, according to the learning rate and their relative responsibility for the error. This backward adjustment of the weights continues all the way to the input layer. After one pass of all the training data, the weights should have been adjusted as to give better classification performance on the next pass. The training process continues until an acceptable number of the training samples are classified correctly. A detailed description of the backpropagation training rule is given in [55].

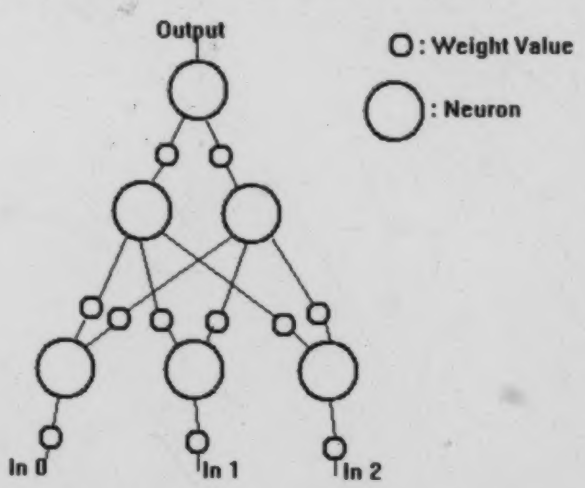

Figure 5-11. A simple 3-2-1 neural net architecture.

Best Number of Neurons Approximation

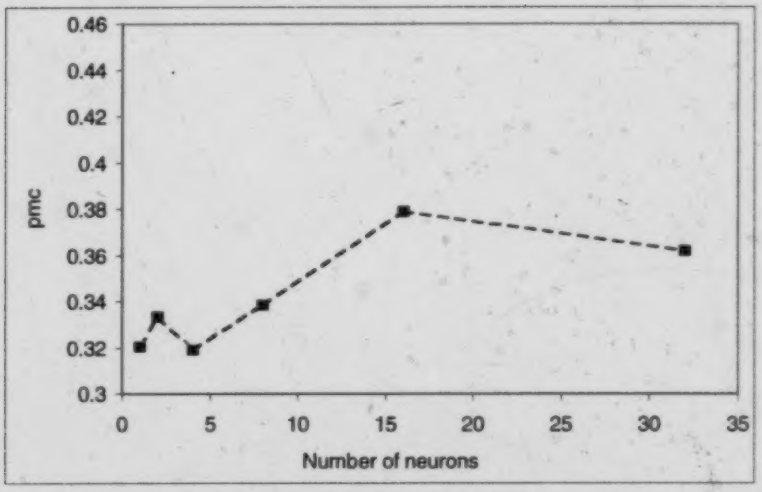

Figure 5-12. Number of neurons in the hidden layer vs. PMC with 16 feature data for subject L.G.

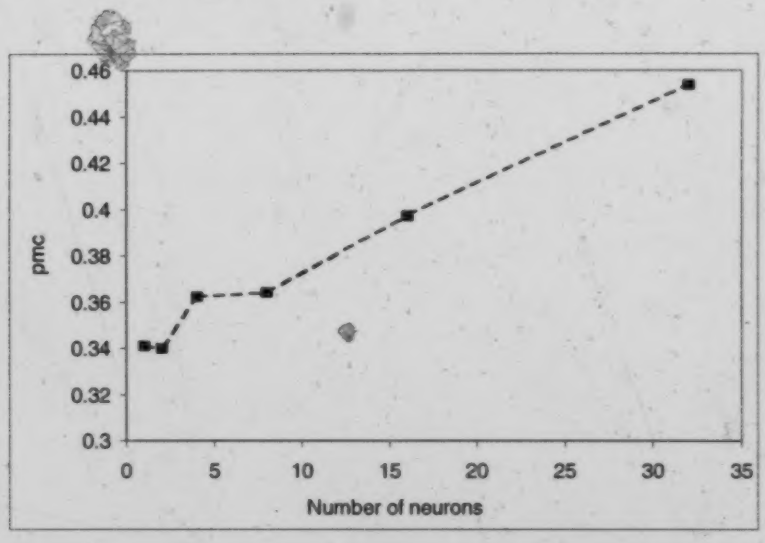

Figure 5-13. Number of neurons in the hidden layer vs. PMC with 32 feature data for subject L.G.

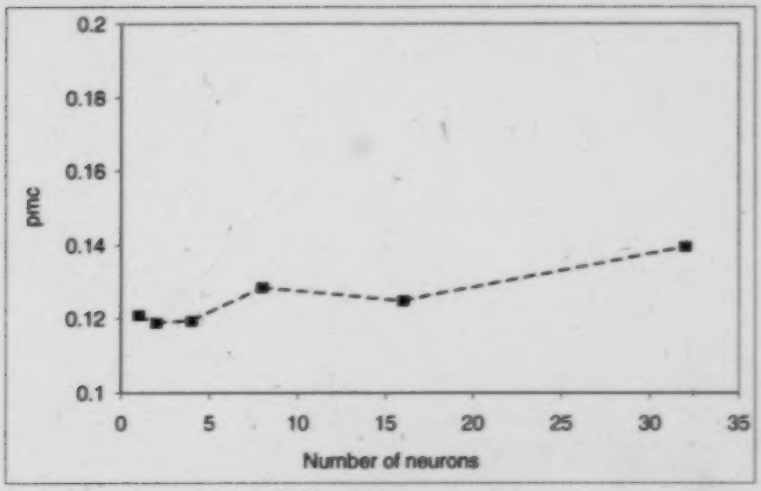

Figure 5-14. Number of neurons in the hidden layer vs. PMC with 16 feature data for subject F.K.

The optimal number of neurons in the hidden layer was approximated by varying the number of neurons while observing the probability of misclassification (PMC) on the reserved data from the training set. The number of neurons in the hidden layer was varied, to try to guess the optimal network size, from 1 to 32 as can be seen in Figure 5-12 and in Figure 5-13 for subject L.G., and in Figure 5-14 and Figure 5-15 for subject F.K. The number of hidden neurons that should give the best results with subject L.G. is four for the 16 feature data set and 2 for the 32 feature data set. The reason for this may be that with 32 features, the input space is less densely sampled than with 16 features and more generalization is needed to get good results. For subject F.K. two neurons predicted the lowest PMC in both the 16 and 32 input space. The weight decay was also adjusted but did not seem to have any noticeable effect on the results.

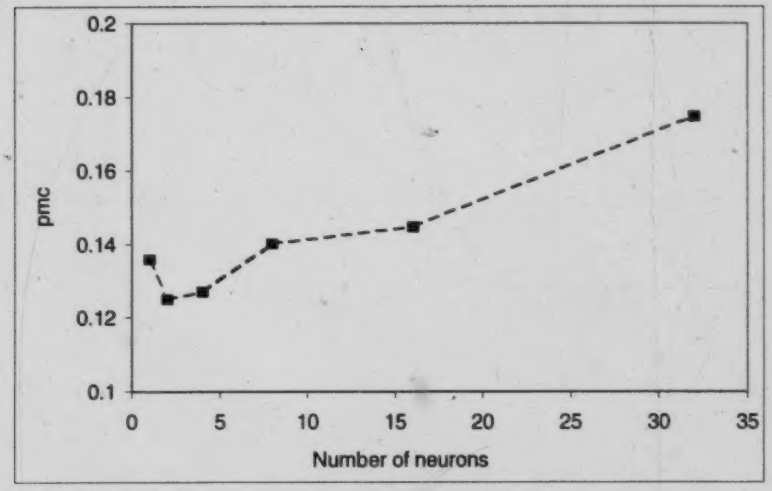

Figure 5-15. Number of neurons in the hidden layer vs. PMC with 32 feature data for subject F.K.

Results

For subject L.G. the median PMC for the BP algorithm tested on the 16 feature evaluation set is 0.3251, and the confusion matrix is shown in Table 5-23. The median PMC for the BP algorithm evaluated on the 32 feature set is 0.3339, and the confusion matrix is in Table 5-24.

		Predicted	
		DOWN	UP
True	DOWN	3420	1981
	UP	1067	2698

Table 5-23. Confusion matrix for MLP-BP on the 16 feature evaluation data set with subject L.G.

		Predicted	
		DOWN	UP
True	DOWN	3305	2096
	UP	1030	2653

Table 5-24. Confusion matrix for MLP-BP on the 32 feature evaluation data set with subject L.G.

For subject F.K. the median PMC for the BP algorithm evaluated on the 16 feature data set is 0.1278, and the confusion matrix is shown in Table 5-25. The median PMC for the BP algorithm evaluated on the 32 feature data set is 0.1325, and the confusion matrix is in Table 5-26.

		Predicted	
		DOWN	UP
True	DOWN	228	4
	UP	67	214

Table 5-25. Confusion matrix for MLP-BP on the 16 feature evaluation data with subject F.K.

		Predicted	
		DOWN	UP
True	DOWN	225	7
	UP	66	215

Table 5-26. Confusion matrix for MLP-BP on the 32 feature evaluation data with subject F.K.

Learning Vector Quantization

Description

Learning Vector Quantization, developed by Kohonen [66] was used by other researchers [24],[25],[27],[28],[29],[31] for EEG classification. Learning vector quantization tries to represent the decision boundaries rather than the class distributions.

LVQ1:

If X is a randomly selected training input vector belonging to some known class and the closest codebook vector C_m to the input has been found by:

$$\| X - C_m \| = \min_k \{\| X - C_k \|\}$$

then the codebook vector is pulled towards X if the classification is correct, or pushed further away from X if classification is incorrect.

The codebook vector C_m is updated according to the rule:

If X and C_m belong to the same class then,

$$C_m^{new} = C_m + \alpha(t) * [X - C_m]$$

If X and C_m belong to different classes then,

$$C_m^{new} = C_m - \alpha(t) * [X - C_m]$$

Where the amount of correction depends on the gain factor $\alpha(t)$, which is a decreasing function of time t.

LVQ2:

An enhanced version of LVQ1 is LVQ2 in which the two closest codebook vectors may get updated. If C_m is the closest codebook vector and C_l is the second closest codebook vector to input X, then C_m and C_l are updated only if the input lies within a specified window around the midplane between the two codebook vectors and the class of X is the same as class of C_l and different than the class of C_m. The updating rule for LVQ2 is:

For the closest codebook vector – different class from X

$$C_m^{new} = C_m - \alpha(t) * [X - C_m]$$

For second closest codebook vector – same class as X

$$C_l^{new} = C_l + \alpha(t) * [X - C_l]$$

All other codebook vectors remain unchanged.

LVQ3:

LVQ2 has a problem that it can lead to a sub optimal equilibrium. A solution is presented in LVQ3 which is an enhanced version of LVQ2. In LVQ3, C_m and C_k are updated if X falls into the window between C_m and C_k and only one of the codebook vectors belongs to the same class as X. Both vectors are also updated if X, C_m and C_k belong to the same class, though the update is not as strong as the first one. The updating rule for LVQ3 is:

For the two closest codebook vectors to X (X is same class as C_l but different class as C_m)

$$C_m^{new} = C_m - \alpha(t) * [X - C_m]$$

$$C_l^{new} = C_l + \alpha(t) * [X - C_l]$$

For the two closest codebook vectors to X (X is same class as C_l and C_m)

$$C_m^{new} = C_m + eps * \alpha(t) * [X - C_m]$$

$$C_l^{new} = C_l + eps * \alpha(t) * [X - C_l]$$

Where *eps* is an extra factor such as 0.2 to make sure that when the classes are the same, the codebook vectors move less than when class of X is the same as class of C_l but different than class of C_m.

Best Number of Reference Vectors

A comparison between LVQ1, LVQ2, and LVQ3 at various codebook sizes can be seen in Figure 5-16. The error comes down in magnitude when four or more codebook vectors are

used; however, it does not seem to improve with a larger codebook beyond the four vectors, so four codebook vectors were used. The median PMC of the evaluation set for codebook of size four with subject L.G. and 16 features data set is 0.3792, 0.3783, and 0.3838 for LVQ1, LVQ2, and LVQ3 respectively, and the confusion matrices are shown in Table 5-27, Table 5-28, and Table 5-29. For the 32 feature data set, a codebook size of four again seems to be the best choice, as can be seen in Figure 5-17. The median PMC for the 32 feature data set is 0.3969, 0.3820, and 0.4297 for LVQ1, LVQ2, and LVQ3 respectively.

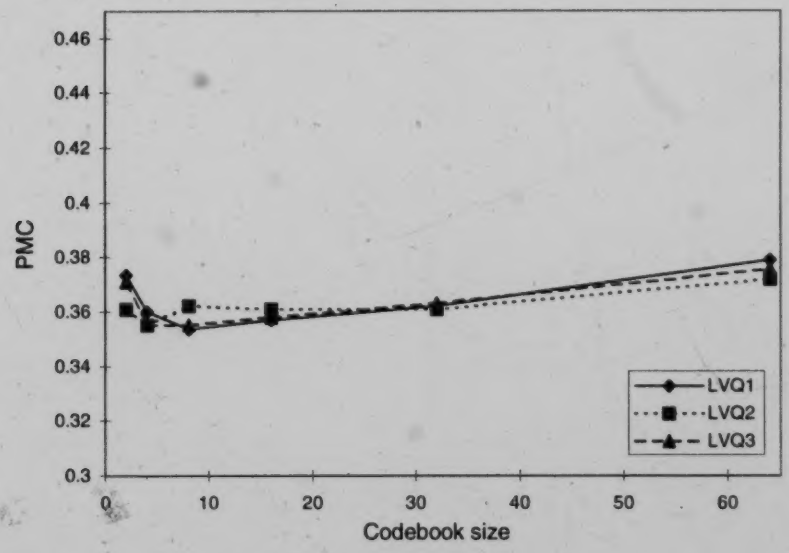

Figure 5-16. Codebook size vs. PMC for LVQ1, LVQ2, and LVQ3 with 16 feature data. Subject = L.G.

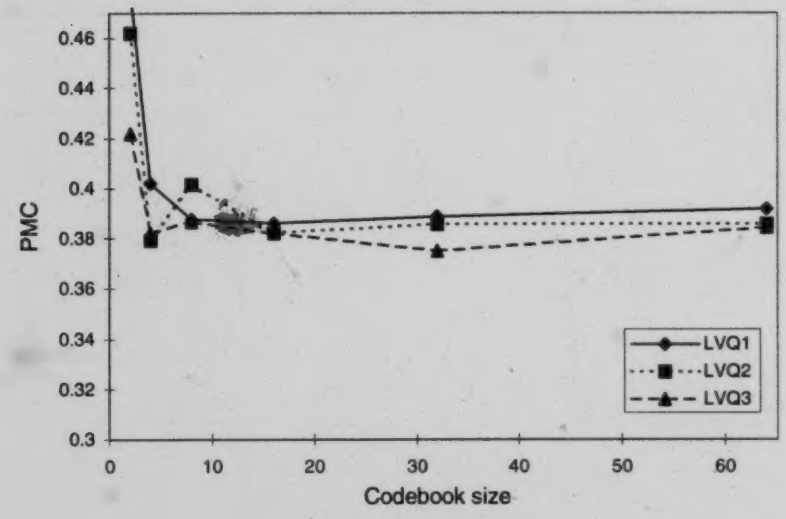

Figure 5-17. Codebook size vs. PMC for LVQ1, LVQ2, and LVQ3 with 32 feature data. Subject = L.G.

		Predicted	
		DOWN	UP
True	DOWN	3010	2391
	UP	1189	2576

Table 5-27. Confusion matrix for LVQ1 on the 16 feature evaluation data. Subject = L.G.

		Predicted	
		DOWN	UP
True	DOWN	3015	2386
	UP	1185	2580

Table 5-28. Confusion matrix for LVQ2 on the 16 feature evaluation data.
Subject = L.G.

		Predicted	
		DOWN	UP
True	DOWN	2990	2411
	UP	1209	2556

Table 5-29. Confusion matrix for LVQ3 on the 16 feature evaluation data.
Subject = L.G.

		Predicted	
		DOWN	UP
True	DOWN	2935	2466
	UP	1270	2495

Table 5-30. Confusion matrix for LVQ1 on the 32 feature test set.
Subject = L.G.

		Predicted	
		DOWN	UP
True	DOWN	3016	2385
	UP	1214	2551

Table 5-32. Confusion matrix for LVQ2 on the 32 feature test set. Subject = L.G.

		Predicted	
		DOWN	UP
True	DOWN	2759	2642
	UP	1394	2371

Table 5-31. Confusion matrix for LVQ3 on the 32 feature data. Subject = L.G.

For subject F.K., a comparison between LVQ1, LVQ2, and LVQ3 at various codebook sizes and a 16 feature data set can be seen in Figure 5-18. Similarly to subject L.G., in general four codebook vectors seem to predict the best results for subject F.K. The median PMC of the evaluation set for a codebook of size four is 0.1532, 0.1572, and 0.1415 for LVQ1, LVQ2, and LVQ3 respectively, and the confusion matrices are shown in Table 5-33, Table 5-34, and Table 5-35. For the 32 feature data set, a codebook size of four again seems to be the best choice, as can be seen in Figure 5- 19. The median PMC for the 32 feature data set is 0.1543, 0.1649, and 0.1515 for LVQ1, LVQ2, and LVQ3 respectively, and the confusion matrices are shown in Table 5-36, Table 5-38, and Table 5-37.

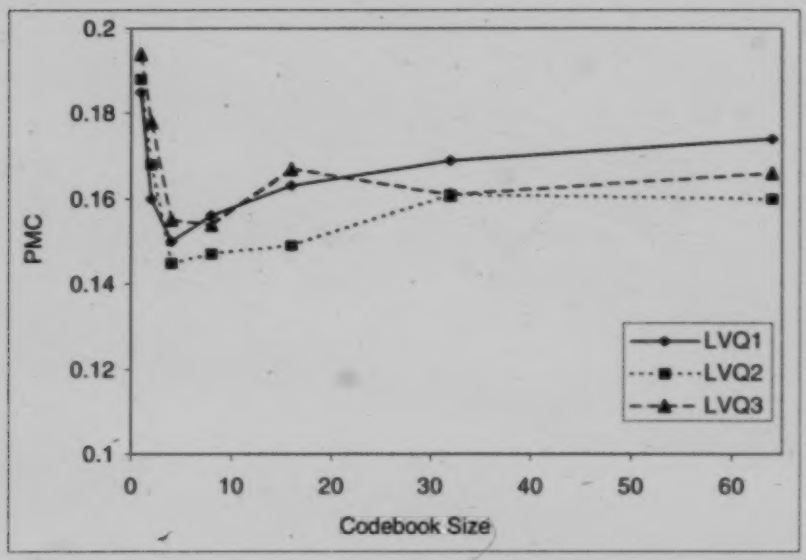

Figure 5-18. Codebook size vs. PMC for LVQ1, LVQ2, and LVQ3 with 16 feature data. Subject = F.K.

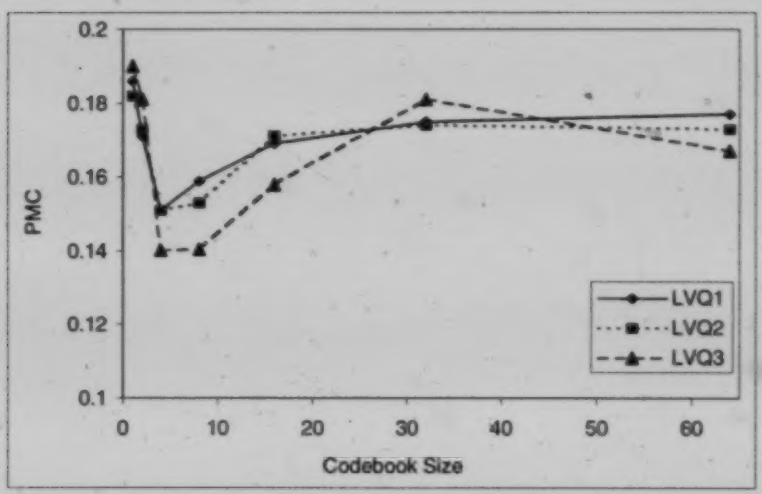

Figure 5- 19. Codebook size vs. PMC for LVQ1, LVQ2, and LVQ3 with 32 feature data. Subject = F.K.

		Predicted	
		DOWN	UP
True	DOWN	222	10
	UP	74	207

Table 5-33. Confusion matrix for LVQ1 on the 16 feature evaluation data. Subject = F.K.

		Predicted	
		DOWN	UP
True	DOWN	221	11
	UP	75	206

Table 5-34. Confusion matrix for LVQ2 on the 16 feature evaluation data. Subject = F.K.

		Predicted	
		DOWN	UP
True	DOWN	220	12
	UP	65	216

Table 5-35. Confusion matrix for LVQ3 on the 16 feature evaluation data. Subject = F.K.

		Predicted	
		DOWN	UP
True	DOWN	205	27
	UP	54	227

Table 5-36. Confusion matrix for LVQ1 on the 32 feature evaluation data. Subject = F.K.

		Predicted	
		DOWN	UP
True	DOWN	217	15
	UP	67	214

Table 5-37. Confusion matrix for LVQ3 on the 32 feature evaluation data. Subject = F.K.

		Predicted	
		DOWN	UP
True	DOWN	210	22
	UP	66	215

Table 5-38. Confusion matrix for LVQ2 on the 32 feature evaluation data. Subject = F.K.

5.3 Discussion of Classifier Performance

As can be seen from Figure 5-20, Figure 5-21, Figure 5-22, and Figure 5- 23, the linear discriminant analysis (LDA), ALN, and multi-layer perceptron with the backpropagation training rule (MLP-BP), in general gave the best results. All three of these pattern recognition systems gave good results but none of them performed best for 16 and 32 feature data sets for both subjects. For example, in the 16 and 32 feature set for subject L.G., LDA gave the best results. For subject F.K., ALN gave the best result in the 16 feature data set, but in the 32 feature data set MLP-BP gave the best results. In all the cases the best ALN results were obtained when the logic tree that was generated was very simple (less than 8 linear pieces). Similarly, backpropagation used 2 or 4 hidden neurons. It seems that the relation being modeled is nearly linear or the data is very noisy. Given the noisy nature of EEG signals, it is expected that simple models do well because they are able to generalize and ignore the noise better than more complex models. Although the K-NN algorithm did not do well compared to the other methods, it performed best with a large number of neighbors (128) which tends to have a smoothing effect.

The relatively poor performance of the LVQ models is a bit of a surprise, since this method is often used in classification of EEG signals. Even with small codebooks (size=4), the LVQ model did not do well compared to the other classifiers. Also, LVQ3, which is the latest and most advanced of the three LVQ models gave worse results than LVQ1 and LVQ2 in both data sets for subject L.G., but better results than LVQ1 and LVQ2 for subject F.K.

Another surprising result is the performance of K-NN algorithm on the 16 feature data set as compared to the 32 feature data set. All other classifiers gave slightly worse results with the 32 feature data than with the 16 feature data, but the K-NN algorithm did just as well. This is interesting because K-NN is a probability density estimation method and should do poorly in high dimensional problems because of the curse of dimensionality [36].

The pattern recognition methods that require the input data to be normalized before being presented to the classifier, such as K-NN and the LVQ algorithms, gave below average results in all the data sets. This observation suggests that the noisy nature of the EEG signals generates some outliers that may cause the normalization process to reduce the resolution in the important regions of the input space, and thus reduce the accuracy with which the classifier can separate the boundaries between the classes

The conclusions that can be drawn from comparing the different classifiers and their observed performance:

- There is no advantage in extracting more than the 16 features from the EEG data using the AR feature extraction method, since performance did not improve significantly by using 32 features as compared to 16 in any of the tested classifiers.
- Simple models that can generalize well give the best performance.
- No single classifier gave the best results for all data sets that were tested; however, ALN, LDA, and MLP-BP gave above average accuracy in all the data sets.
- The classification function appears to be slightly nonlinear since the nonlinear classifiers did not give best results when trained with linear parameters. For example, MPL-BP did not predict best results with only one hidden neuron, or ALN with one linear piece, or LVQ with only two codebook vectors.

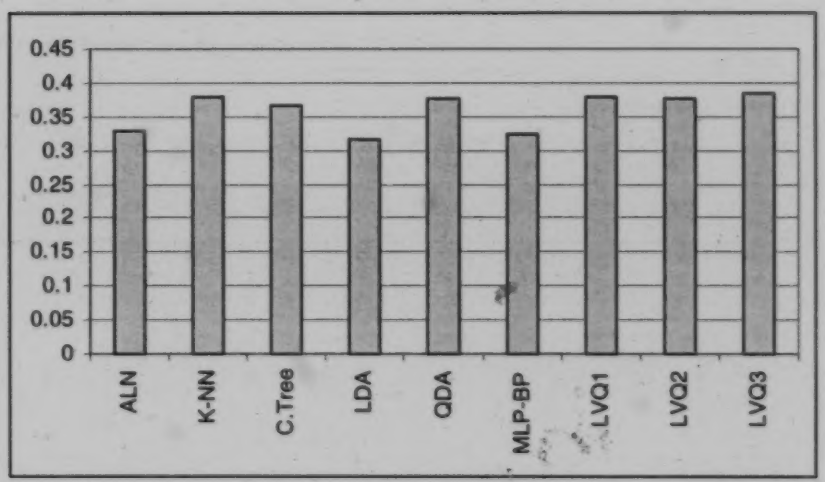

Figure 5-20. Comparison of PMC with different classifiers on the 16-feature data set for subject L.G.

The real-time BCI training system requires an adaptive classifier that can learn as training data becomes available. Because an adaptive classifier is required, linear discriminant analysis is not acceptable. An adaptive version of linear discriminant such as a single perceptron would meet the adaptive criteria; however, as was seen in MLP-BP with one

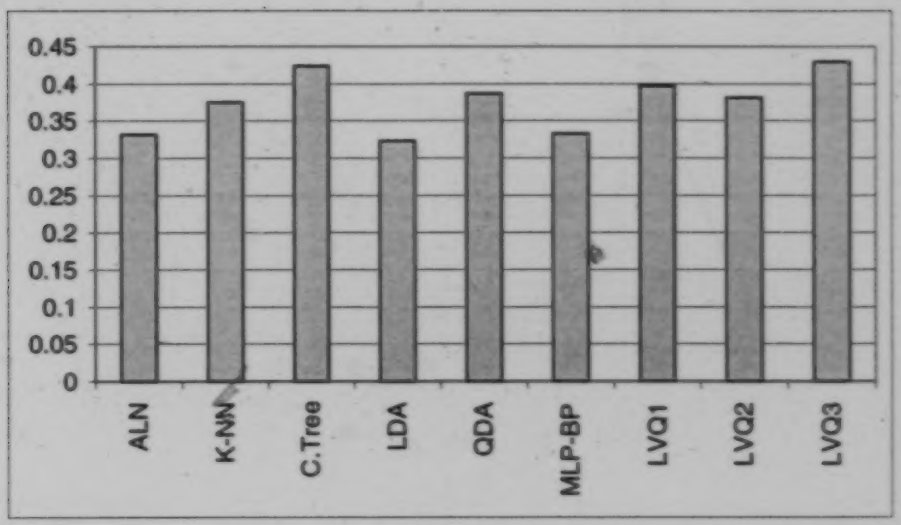

Figure 5-21. Comparison of PMC with different classifiers on the 32-feature data set for subject L.G.

hidden neuron, the results are not as high. MLP-BP would be an acceptable classifier in terms of accuracy; however, it is relatively slow (requires high computational power) in training and testing. ALN can give comparable accuracy at a fraction of the computational time, thus making it a better choice for a classifier in the real-time BCI system, where multiple classifiers may have to be trained and tested in parallel.

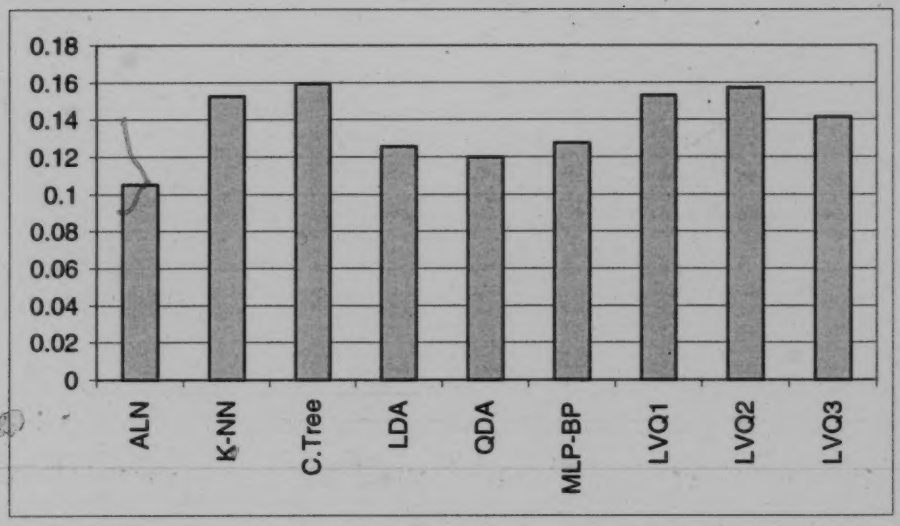

Figure 5-22. Comparison of PMC with different classifiers on the 16-feature data set for subject F.K.

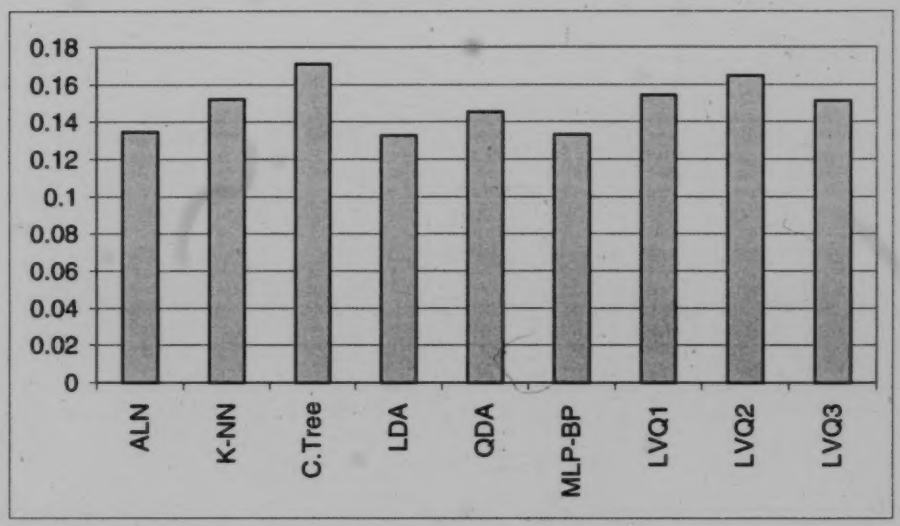

Figure 5- 23. Comparison of PMC with different classifiers on the 32-feature data set for subject F.K.

5.4 Ensembles of Classifiers

It has been shown [56] that an ensemble of classifiers whose individual decisions are combined in some way to classify new examples can give improved results. Ensembles are often much more accurate than the individual classifiers that make them up. However, an ensemble of classifiers can be more accurate than its component classifiers only if the individual classifiers disagree with one another [57]. For example, if a classifier has an error rate of 30%, then taking an ensemble of just three uncorrelated classifiers each with an error rate of 30% and performing unweighted voting will produce a reduced error rate of 21.6%. Using an ensemble of five uncorrelated classifiers each with the same 30% error rate should reduce the effective error rate to just 16.3%. Unfortunately, if the errors made by the classifiers are correlated then when one classifier is wrong then all of the classifiers will be wrong and no enhancement in accuracy can be made.

Many methods for constructing ensembles have been developed. One method is to manipulate the training samples to generate multiple classification functions. The learning algorithm is run several times, each time with a different subset of the training samples. One of the simplest methods of manipulating the training set is called bagging. Bagging presents each classifier with a training set that consists of a sample of N training examples drawn randomly with replacement from the original training set of N samples. Each re-sampled copy of the training set contains, on the average, 63.2% of the original training set, with several training examples appearing multiple times. It is hoped that each new re-sampled training set will produce a slightly different classification function when presented to a classifier. If each classification function is adequately different, it is anticipated that when the classifiers are polled, the majority vote will give the correct answer more often than a single classifier.

Since ALN is an extremely fast pattern recognition system, in training and evaluation, it may be advantageous to construct an ensemble of several ALNs to vote on a response to each testing sample. Two methods of constructing an ensemble have been tested and compared to classification with a single ALN. The first method is to use the entire training set with five ALNs which are randomly initialized to train the five classifiers. In this method, it is hoped that each ALN will give a slightly different classification surface

due to the random initialization. The second method is to create five separate training sets with the bagging technique, and then train on these data sets with five randomly initialized ALNs. The advantage of the bagging method is that the randomly initialized classifiers should produce less similar functions and are more likely to be uncorrelated than by using the full data set with randomly initialized classifiers. The disadvantage of this method is that a fraction of the samples are always left out of training, which may produce negative results if the size of the training set is small with respect to the classification problem. Figure 5-24 and Figure 5-25 show the results of comparing the two described methods of creating classifier ensembles with just one ALN. The method labeled "One ALN" is just one ALN trained on the entire training set and evaluated on a separate testing set. The method labeled "Voting" used the entire training set with five randomly initialized ALNs and for each sample in the testing set took an unweighted vote of these five ALNs. The method labeled "Bagging and Voting" created five separate training sets according to the bagging technique, and used them with five randomly initialized ALNs and for each sample in the testing set took an unweighted vote of these five ALNs. The entire procedure was repeated five times and the median PMC taken for the "One ALN", "Voting", and "Bagging and Voting" methods, in order to reduce any possible random variation between the runs. Although the most practical number of AR features per sample was analyzed in chapter 3, the three methods described above were tested with 2, 4, 8, and 16 AR features per lead so that it is possible to observe the effect of different dimensionality on each of these methods.

As can be seen from Figure 5-24 and Figure 5-25, using only one ALN consistently gives higher error than the other two methods, at any number of features and for both subjects, even though the difference in error rate may not always be significant. The difference between the "Voting" and "Bagging and Voting" methods in terms of performance is inconsistent and insignificant (usually less than 1%). This result may indicate that the random initialization makes the classification functions uncorrelated enough to give better results than just one classifier, but random re-sampling either does not help, or else any positive effects of the re-sampling are negated by using fewer unique training samples.

Using ALNs as regression approximants and averaging the outputs of five ALNs, with bagging, was also attempted. The difference in the results between applying ALNs as

regression approximants with averaging, versus applying ALNs as classifiers with polling was not statistically significant.

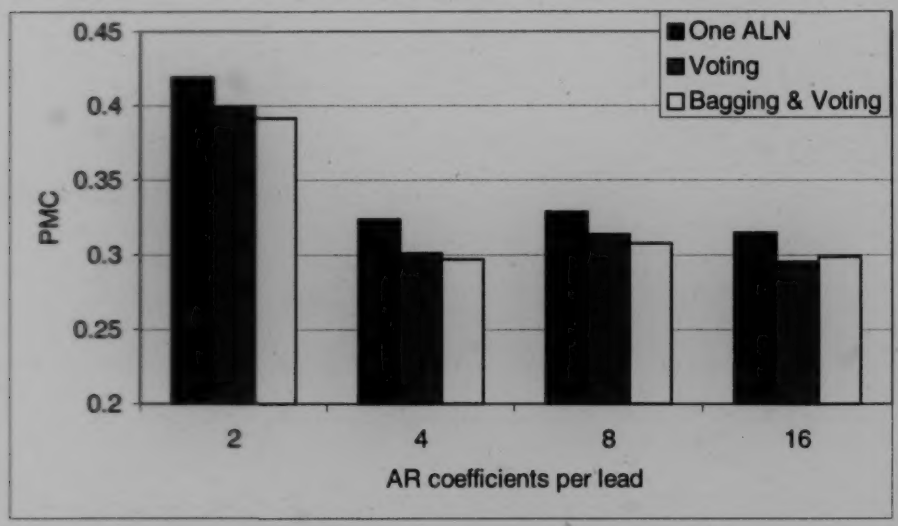

Figure 5-24. Comparison of PMC with "One ALN", "Voting", and "Bagging & Voting" techniques for different number of AR coefficients. Subject = L.G.

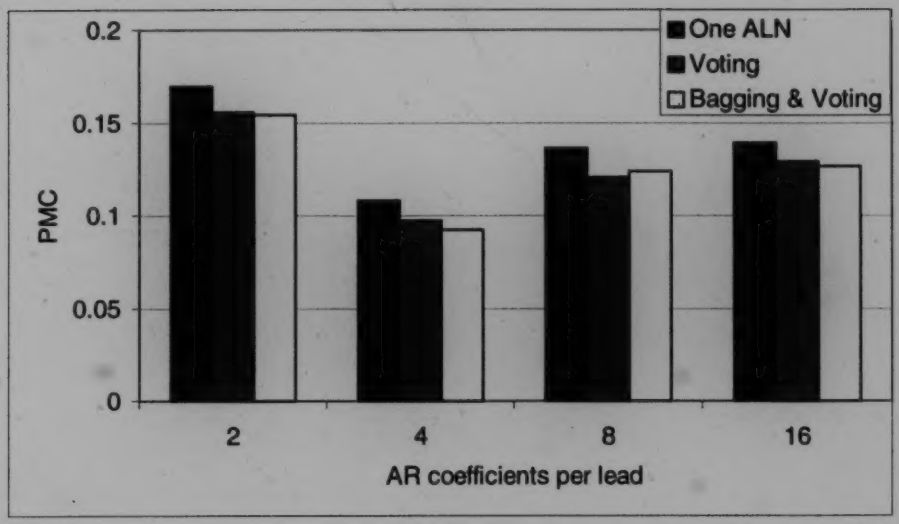

Figure 5-25. Comparison of PMC with "One ALN", "Voting", and "Bagging & Voting" techniques for different number of AR coefficients. Subject = F.K.

Chapter 6

Results

In this chapter the results of training with various subjects are summarized for one and two dimensional cursor control. As was previously noted, our subjects were volunteers and thus their participation was scheduled according to their convenience. As a result the training of the subjects was not as consistent and intense as would have been optimal.

6.1 One Dimensional Cursor Control

Subjects

Six subjects stayed with the study long enough to achieve at least some one-dimensional control of the cursor on a computer screen. If no sign of cursor control was shown during the training phase of the session, an evaluation was not done. If some control was observed during training, a performance evaluation was carried out at the end of the session at various cursor speeds. The subject was given ten randomly selected targets (half UP and half DOWN) and was asked to move the cursor towards the target. The test set of ten targets was given at various cursor speeds during the evaluation phase of each session. A speed where it takes a minimum of 32 steps to reach the target at 0.05 seconds per step was considered to be practical for applications, due to high accuracy and fast speed, so the results for each subject are shown at this cursor speed.

Subject J.K.

J.K. is a middle age man suffering from progressive multiple sclerosis. It took J.K. about five sessions to show signs of control during training. J.K. stayed for twelve more sessions after which he still showed only marginal one-dimensional cursor control. It is not certain why better control was not obtained and whether a more intense or longer participation would have helped. A summary of the results obtained during the evaluation phase can be seen in Figure 6-1.

Subject L.U.

L.U. is a healthy middle-aged woman. L.U. was able to achieve some control in her second session. She stayed for five more sessions after which she still showed limited one-dimensional cursor control. A summary of the results obtained during the evaluation phase can be seen in Figure 6-2.

Subject R.J.

R.J. is a healthy man and a recent university graduate. R.J. was able to achieve good control in his first session. He stayed for a total of seven sessions. A summary of the results obtained during the evaluation phase can be seen in Figure 6-3. In his fifth session R.J. was tired and had trouble concentrating on the task.

Subject F.K.

F.K. is a healthy man in his early twenties, and a university student. F.K. was able to achieve good control in his second session. He did only two more one-dimensional sessions in both of which he was able to hit the target 100% of the time during evaluation, and afterwards helped with the real-time testing of several BCI applications during development.

Subject G.M.

G.M. is a middle age man suffering from post-polio. G.M. was able to achieve one-dimensional cursor control in his third session and has done ten more one-dimensional sessions since then, as can be seen in Figure 6-4.

Subject L.G.

L.G. is a healthy man in his late twenties and a recent university graduate. L.G. was able to achieve one-dimensional cursor control in his fourth session after which he did twenty-one more one-dimensional sessions, as can be seen in Figure 6-5.

Conclusions from One-Dimensional Control Sessions

It has been our observation that anyone can achieve some one-dimensional control of the cursor on the screen using BCI. The number of sessions required before a subject can achieve some control varies from person to person, but in general is less than six half-hour sessions. We have not been able to generalize the reason why some people seem naturally

more adept to modulate their EEG signals. If the subject continues training sessions after control was first achieved, he will require less focus to get the same results. The averaged spectra for each subject during one-dimensional cursor control experiments are included in the Appendix.

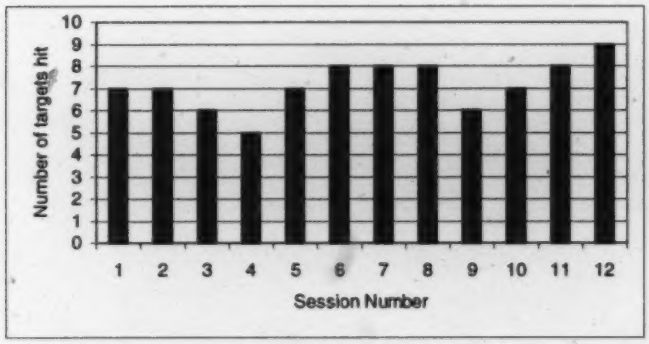

Figure 6-1. One-dimensional real-time evaluation results for J.K.

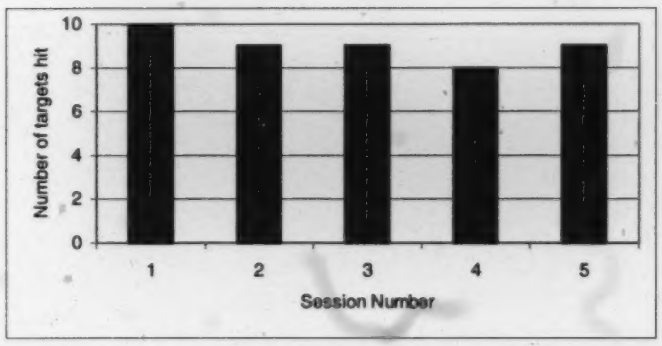

Figure 6-2. One-dimensional real-time evaluation results for L.U.

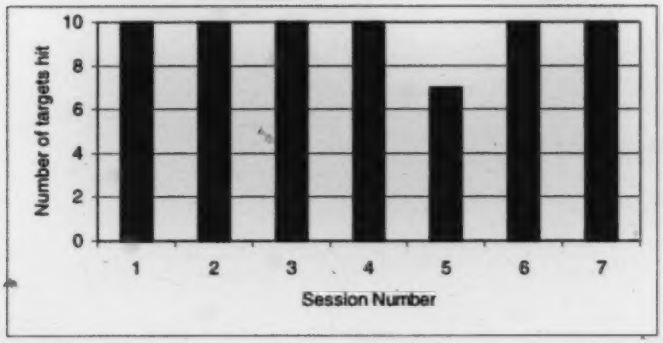

Figure 6-3. One-dimensional real-time evaluation results for R.J.

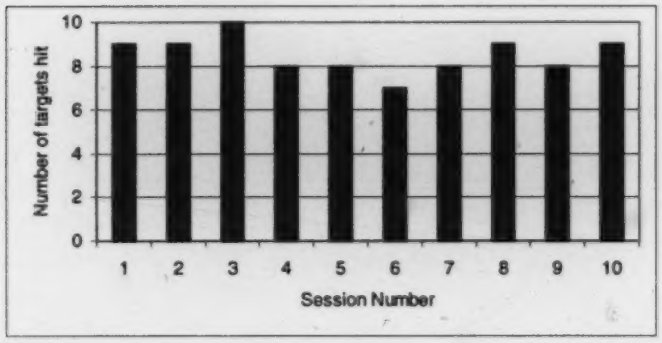

Figure 6-4. One-dimensional real-time evaluation results for G.M.

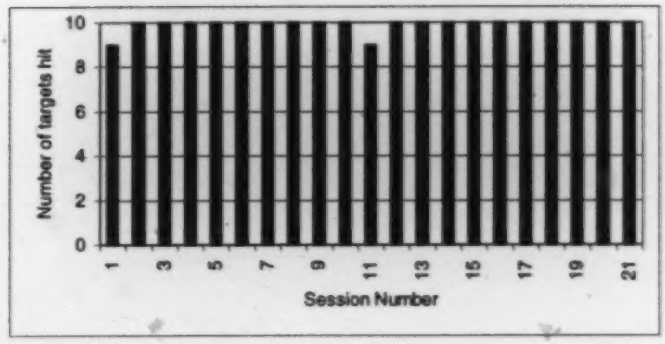

Figure 6-5. One-dimensional real-time evaluation results for L.G.

6.2 Two Dimensional Cursor Control Results

Two dimensional cursor control (UP, DOWN, LEFT, and RIGHT) is much more difficult than one-dimensional control. We only attempted to train a subject for two-dimensional control after he showed excellent one-dimensional performance. We attempted two-dimensional control with three of our subjects. Subject F.K. attempted to learn two-dimensional control for three sessions during which he was not successful and did not wish to try further. Subjects L.G. and G.M. both got some two-dimensional control after less than five more sessions each. After these two subjects achieved some two-dimensional control, one-dimensional training was not generally continued. Two-dimensional cursor control results can be seen in Figure 6-6 and Figure 6-7 for subjects G.M. and L.G. respectively. It should be noted that in two-dimensional evaluation, any lack of control should result in a performance of 25%. During two-dimensional control evaluation, performance was tested only on one cursor speed (32 steps to target). In general, more time is required for the training phase in two- versus one-dimensional training. As a result, the evaluation phase at times had to be stopped early as the subject was starting to feel fatigued.

The accuracy does not seem to improve, in general, with more training sessions after the subject first achieved two-dimensional control. A possible explanation may be that a more intense BCI training schedule would be required to further improve accuracy. The

averaged spectra for each subject during two-dimensional cursor control experiments are included in the Appendix (Figures A-7 and A-8).

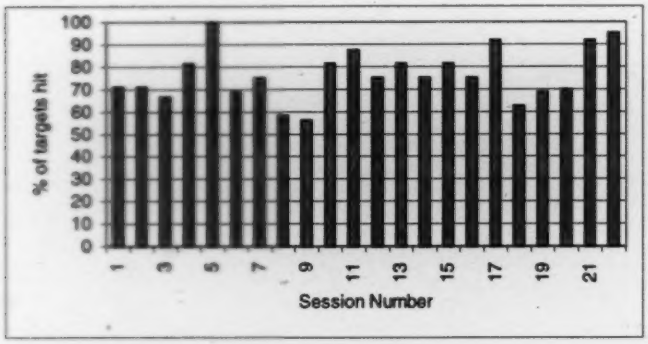

Figure 6-6. Two-dimensional real-time evaluation results for G.M.

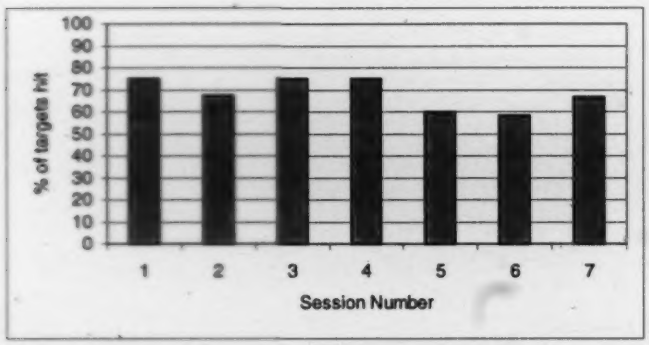

Figure 6-7. Two-dimensional real-time evaluation results for L.G.

Chapter 7

BCI Applications

In this chapter we describe the applications that were developed using our real-time BCI method. These applications were developed to be used with one-dimensional control because it gives better accuracy and is easier to learn.

7.1 Training Setup for Control of Neural Prosthesis

In this section we report on the development of experimental setup and initial results for control of an artificial wrist through a brain computer interface (BCI). As a first step towards the development of thought-controlled prosthesis, we created a system that allows a user to move an animated wrist on a computer screen in the up and down directions. In our experiments, we used surface Electroencephalogram (EEG) signals above sensory-motor areas, while the subjects were attempting to use only mental activities to modulate their EEG signals resulting in desired movements of the animated wrist. In our initial trials with two subjects, both demonstrated the ability to move the animated wrist as desired. These results provide a good basis for developing more enhanced wrist controls such as turn, grasp and release.

Introduction

Loss of function in the upper extremity, as a result of an accident or a disease, is one of the more severe disabilities that people can sustain [58]. Without the use of upper extremities, even simple manipulative functions that are necessary for basic daily tasks must be replaced by alternative assistive measures. However, the effectiveness of most assistive devices is dependent on preserved residual movements or speech. Without any physical channels for control, the only alternative for these people may be in exploring indirect voluntary modulation of electrical fields resulting from neural processes in their brains.

Figure 7-1. A picture of the animated wrist in a centered position with the desired direction of movement.

Methods

The subject is comfortably seated in front of a monitor while EEG signals are analyzed in real time from leads C3 and C4. The subject's goal is to keep changing the angle of an animated wrist on the computer screen in an up or down direction until the upper or lower edge of the screen is touched. An arrow appears on the subject's screen indicating the desired direction of movement, as can be seen in Figure 7-1. Each of the experimental sessions lasts approximately 30 minutes. The first half of each session is used to train a new classifier and the second half is used to evaluate the performance. In the training and evaluation of each subject, auto regressive coefficients are used as features to an artificial neural network classifier.

Results

We used two subjects in this study who have been trained with our BCI system for over 10 sessions each. One subject was non disabled, while the other had Post-Polio syndrome. Both subjects were able to successfully control the animated wrist using BCI, and were able to touch the lower and upper edge of the screen with 100% accuracy during evaluation.

Discussion

The goals of an upper extremity neural prosthesis are directed toward establishing more independence for the user. A BCI controlled prosthesis has the advantage of being usable by patients with the most severe disabilities and can be controlled without any invasive procedures. Our system is still under development for real life applications; nevertheless, the initial results with our two subjects are positive. The challenge that remains is to increase the capabilities of the BCI controlled prosthesis while transferring it into clinical applications.

7.2 Environmental Control by a Brain-Computer Interface

People with disabilities, such as those with amytrophic lateral sclerosis (ALS), lose control of their bodies, leaving them unable to perform simple tasks such as speech, locomotion, and the ability to effectively interact with their environment. Brain-Computer Interfaces (BCI) show promise in allowing these individuals to interact with a computer using EEG. A system was created to allow individuals, via a BCI, to control home appliances and to have the computer articulate pre-selected words. This setup uses ActiveHome from X10 (X10 Ltd. Product Information - accessible through www.x10.com), which allows control of a wide variety of home appliances via a computer, and executes tasks such as light dimming, channel changing, and turning appliances on and off. The system was evaluated with a set of tasks to measure its ease-of-use, ease-of-learning, rate-of-error, and amount of time the subject required to complete the task.

Environmental Control Background

A variety of work has been done for people with severe disabilities to control their environment. Most require some neuromuscular control. For example, home automation, often via X10, has been demonstrated utilizing voice, switch [59], and eye twitch [60] sensors. Enlarged keyboards such as Intellikeys [61] allow typing with limited motor control.

Work has also been undertaken for patients who have minimal or no neuromuscular control. Direct control of a single on/off switch using EEG alpha rhythms has been achieved [62]. Control of an on-screen keyboard has been achieved using a single electrode implanted in the human motor cortex of the brain [64]. Implanted electrodes in primates have

controlled prosthetic devices [63]. Another system allows subjects to parse a binary menu tree, selecting between yes and no answers at each level using a BCI [45].

The system described in this section, named Environmental Control-BCI (ECBCI) is controlled by the BCI. It is designed to perform a variety of tasks in a reduced timeframe. ECBCI provides four choices at each menu level, including one choice to move to the previous menu. It is designed to work with a wide variety of applications, and to add new applications as they become available.

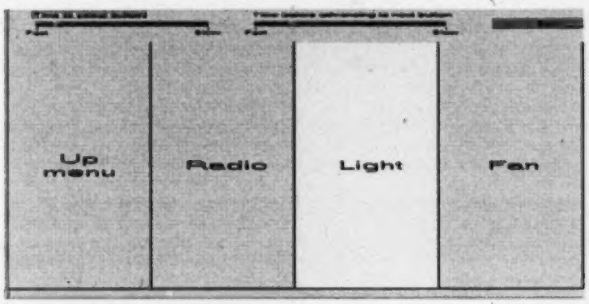

Figure 7- 2. A submenu of ECBCI

Methods

User Interface

The user interface of ECBCI, developed under Windows, consists of four vertical buttons, with the first button usually labeled "Up Menu" to return to the previous menu, as can be seen in Figure 7-2. A scanning algorithm is employed, which uses two BCI states: "scan" and "stop". When the "scan" state is detected, each button is highlighted for a certain period of time before the system scans to the next button. The entire button is highlighted to ensure subjects with poor vision are likely to see it. When the BCI "stop" state is detected, the scanning stops, and the current button remains highlighted. If the button remains highlighted for a specified time, it is selected. A selected button can either open a submenu, or perform an action such as activating appliances or playing a sound. The amount of time before a button is selected is configurable.

ECBCI contains a lock-mode, which effectively "locks" the system and prevents the user from accidentally selecting a menu item when they are not concentrating on the BCI. In lock-mode, control of the system is blocked until a specific sequence of four buttons is selected, as is shown in Figure 7-3. Once the four buttons are correctly chosen in sequence, the top menu returns. If an incorrect button is selected while unlocking, the subject must

start again from the first step. Lock-mode is activated by selecting "Up Menu" from the top menu.

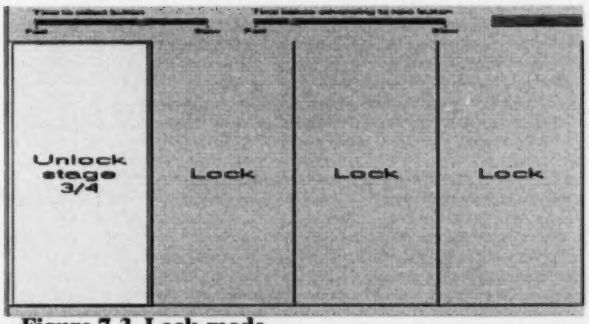

Figure 7-3. Lock-mode

A sample menu tree was developed based on consultation with rehabilitation professionals. It encompasses a cross-section of the tasks an individual may wish to perform. It contains 10 submenus and 19 action buttons (Figure 7-4).

A sequence of actions was developed to evaluate the speed and number of errors made by subjects using ECBCI. It contains 12 actions requiring a total of 44 button presses. The actions in the sequence are "Hello", "How Are You?", Turn radio on, Set light to high, Turn radio off, "I am hungry", Set light to low, Turn fan on, "I am tired", Turn light off, Turn fan off, and Lock-ECBCI.

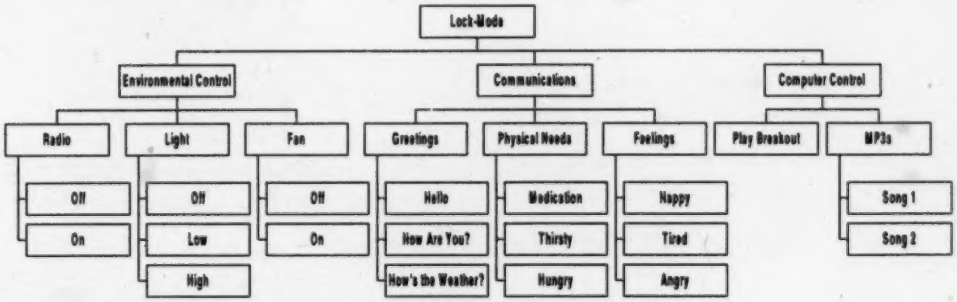

Figure 7-4: The Menu Structure Developed for Evaluation

Environmental Control

ECBCI is designed to be flexible enough to accommodate a wide variety of actions, from controlling home appliances and prosthetics, to articulating pre-recorded phrases, playing music and starting games. ECBCI is designed to be distributed across multiple machines, so

that processing of EEG is not hindered by other tasks, such as sound playback. It is highly configurable so that external software can be executed, to allow for easy design of different menu structures. A few modes of action have been developed and tested (Figure 7-5) to control home appliances, to articulate phrases by playing sounds and to start games.

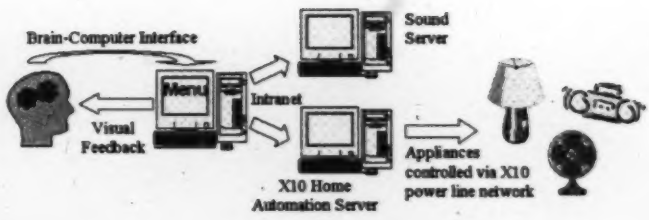

Figure 7-5. ECBCI System Configuration

In the evaluation configuration there is a radio and a fan that can be switched on and off, and a lamp that can be set to various levels of brightness. The X10 system sends control signals through power-lines using a high-frequency modulated carrier. The transmitter connects to the computer serial port. The radio and fan are controlled by a Universal Appliance Switch and the incandescent lamp by a Dimmer Switch, both of which are commonly available. The X10 software includes a command-line program to transmit instructions to appliances, which is used by ECBCI.

Windows Media Player is used to articulate phrases, which are recording as wave sound files. Games developed for the BCI were successfully started from ECBCI.

There is an endless choice of other applications, since any software that can be executed from the command-line can be initiated by ECBCI. This results in a flexible architecture.

Results

The scanning algorithm used worked extremely well, and the test subjects were able to carry out various sequences of tasks quickly and with minimal error.

Preliminary trials were conducted on a healthy test subject with a sequence containing 12 actions that required 44 button presses to navigate the menu tree, as can be seen in Table 7-1. None of the errors made were "serious," in that they did not communicate any messages that would affect the user, nor did they change the state of any devices the user did not want to affect. Qualitative tests were also performed with a post-polio subject. He was able to successfully complete the sequence.

Trial	Time required (min:sec)	Mean time per action	Mean time per button	Actions erroneously selected
1	5:00	25.0 sec	6.8 sec	2
2	6:05	30.4 sec	8.3 sec	1
Mean	5:33	27.7 sec	7.6 sec	

Table 7-1. Preliminary Results From Subject Trials

Lock-mode did not adequately meet its objective of locking ECBCI. When a test was performed for accidental unlocking, the subject, who could not see the screen, unlocked and then relocked ECBCI in about five minutes. Calculations of the probability that the sequence could accidentally be selected in a certain amount of time were made, and they confirmed the observation that lock-mode is not adequate.

The X10 controller does not always function reliably. Communication is not possible between certain combinations of electrical sockets within the lab. This is likely due to the sockets in the building using a three-phase system and thus some combinations are out of phase. In addition, some power bars impede signal transmission, while others do not. Commands sent from the transmitter are not always received; especially by a receiver that has just been plugged in and which has not had time to "warm-up". Devices that are told to turn off sometimes turn off and then immediately turn back on. X10 attributes all of these problems to power line interference. Home automation using methods other than X10 to transmit the BCI commands can be implemented. Despite these issues, the system works satisfactorily if the correct configuration is found.

Discussion

The scanning mode is the most effective method found to control the user interface. The software has been written with an open architecture that allows an endless number of existing and new applications. It has been successfully tested with the X10 system, with sound files, and to start computer applications.

Initial subject evaluations show promising results. The subject was able to select buttons in less than eight seconds, and actions in less than 30 seconds. This is comparable to

the amount of time it takes to get up from ones' computer, go to the radio, turn it on and return to the computer. The subject was able to navigate the menus with few errors. The lock-mode works, but may not provide adequate security; a longer sequence may be required.

Further testing with multiple subjects is required to confirm the usefulness of ECBCI, and to find the optimal configuration that minimizes the time required to perform actions and the number of errors. Investigation is required into the effects of cursor velocity, the number of buttons on the screen, the mode of cursor movement used and the time required to select a button.

Chapter 8

Conclusion

The effective control of advanced assistive devices such as neuroprostheses, environmental control units, and communication aides have many common problems that may limit their clinical application. Many of these can be directly attributed to an inadequate man-machine interface. We started with the hypothesis that human subjects with severe neuromuscular disorders can be trained to modulate their brain electrical activity and to signal their simple intentions through the extracranial electroencephalogram, and we designed a communication and control system for assistive devices based on a simple brain-computer interface. By using parallel human-machine training and focusing on the pattern recognition technology, we have shown that it is possible to train some subjects to move a cursor in one dimension on a screen and be able to hit the target 100% of the time. Two dimensional cursor control has also been achieved, but with limited accuracy.

Our next phase of research should focus on the following three areas:

1. Determine if a secondary task of varying complexity can be performed at the same time as BCI control. Controlling the cursor while performing other mental tasks is of crucial importance for practical application of this technology in order to insure that a subject can use their EEG control interface for functional tasks. Data reported qualitatively by Wolpaw and McFarland [9] about well-trained users, who can control the cursor while carrying on a casual conversation or while performing mental arithmetic problems and giving the answers, are very promising but not fully quantified. It is necessary to assess the capability of the subjects objectively to perform a secondary task during cursor control at different skill levels.
2. Determine if the same detection possibility exists with at least five patients having long standing sensorimotor system impairment.
3. Determine if a long-term and intense subject-training schedule gives significantly higher accuracy in two dimensional cursor control.

Bibliography

[1] G. Pfurtscheller, D. Flotzinger, W. Mohl, and M. Peltoranta. Prediction of the side of hand movements from single-trial multi-channel EEG-data using neural networks. *Electroencephalography and clinical Neurophysiology.*, vol. 82, pp. 313-315, 1992.

[2] G. Pfurtscheller, C. Neuper, A. Schloegl, and K. Lugger. Separability of EEG signals recorded during right and left motor imagery using adaptive autoregressive parameters. *IEEE Transactions on Rehabilitation Engineering*, vol. 6, no. 3, pp. 316-325, 1998.

[3] J.R. Wolpaw, D.J. McFarland, G.W. Neat, C.A. Forneris. An EEG-based brain-computer interface for cursor control. *Electroencephalography and clinical Neurophysiology.*, vol.78, pp. 252-259, 1991.

[4] J.R. Wolpaw and D.J. McFarland. Multichannel EEG-based brain-computer communication. *Electroencephalography and clinical Neurophysiology.*, vol. 90, pp. 444-449, 1994.

[5] T.M. Vaughan, J.R. Wolpaw, and E. Donchin. EEG-Based Communication: Prospects and Problems. *IEEE Transactions on Rehabilitation Engineering*, vol. 4, no.4, pp. 425-430, 1996.

[6] G. McMillan and G. Calhoun. Direct brain interface utilizing self-regulation of steady-state visual evoked response (SSVER). *RESNA'95*, pp.693-695, 1995.

[7] E.E. Sutter. The brain response interface: communication through visually guided electrical brain responses. *Journal of Microcomputer Applications*, vol. 15, pp. 31-45, 1992.

[8] L.A. Farwell and E. Donchin. Talking off the top of your head: Toward a mental prosthesis utilizing event-related brain potentials. *Electroencephalography and clinical Neurophysiology*, vol. 70, pp. 510-523, 1988.

[9] J.R. Wolpaw and D.J. McFarland. Development of an EEG-based brain-computer interface (BCI). *RESNA'95*, pp.645-649, 1995.

[10] J.J. Vidal. Toward direct brain-computer communication. *Annual Review of Biophysics*, pp. 157-180, 1973.

[11] T. Elbert, B. Rockstroh, W. Lufzenberger, and N. Birbaumer. Biofeedback of slow cortical potentials. *Electroencephalography and clinical Neurophysiology*, vol. 48, pp. 293-301, 1980.

[12] B. Rockstroh, T. Elbert, A. Canavan, W. Lutzenberger, and N. Birbaumer. *Slow Cortical Potentials and Behaviour.* 2nd edition, Urban and Schwarzenberg, Baltimore, MD, pp. 127-150, 1989.

[13] Z.A. Keirn and J.I. Aunon. Man-machine communication through brain-wave processing. *IEEE EMB Mag.*, vol. 9, pp.55-57, 1990.

[14] T.A. Travis, C.Y. Kondo, and J.R. Knott. Alpha enhancement research: a review. *Biol. Psychiat.*, vol. 10, pp. 69-89, 1975.

[15] W.N. Kuhlman. EEG feedback training: enhancement of somatosensory cortical activity. *Electroencephalography and clinical Neurophysiology*, vol. 45, pp. 290-294, 1978.

[16] A.H. Black, G.A. Young, and C. Batenchuk. Avoidance training of hippocampal theta waves in flaxedilized dogs and its relation to skeletal movement. *J. Comp. Physiol. Psychol.*, vol. 70, pp. 15-24, 1970.

[17] M.D. Craggs. The cortical control of limb prostheses. Ph.D. thesis, University of London, U.K., 1974.

[18] E. Niedermeyer and F.H. Lopes da Silva. *Electroencephalography: Basic Principles, Clinical Applications and Related Fields*. Urban and Schewarzenberg, Baltimore, MD, pp.97-117, 1993.

[19] G. Pfurtscheller. Central beta rhythm during sensorimotor activities in man. *Electroencephalography and clinical Neurophysiology*, vol. 51, pp. 253-264, 1981.

[20] J.B. Polikoff, H.T. Bunnell, and W.J. Borkowski. Toward a P300-based computer interface. *RESNA '95*, pp. 678-680, 1995.

[21] D. Patmore and R.B. Knapp. A cursor controller using evoked potentials and EOG. *RESNA '95*, pp. 702-704, 1995.

[22] G. Pfurtscheller, D. Flotzinger, and J. Kalcher. Brain-Computer Interface - a new communication device for handicapped persons. *Journal of Microcomputer Applications*, vol. 16, pp. 293-299, 1993.

[23] M. Pregenzer and G. Pfurtscheller. Frequency component selection for an EEG-based brain to computer interface. *IEEE Transactions on Rehabilitation Engineering*, vol.7, no. 4, 1999.

[24] G. Pfurtscheller, J. Kalcher, C. Neuper, D. Flotzinger, and M. Pregenzer. On-line EEG classification during externally-paced hand movements using a neural-network-based classifier. *Electroencephalography and clinical Neurophysiology*, vol. 99, pp.416-425, 1996.

[25] J. Kalcher, D. Flotzinger, G. Pfurtscheller, S. Golly, and C. Neuper. Graz brain-computer interface II: Studies for an EEG-based communication device. *RESNA '95*, pp.690-692, 1995.

[26] G. Pfurtscheller and A. Berghold. Pattern of cortical activation during planning of voluntary movement. *Electroencephalography and clinical Neurophysiology*, vol. 72, pp. 250-258, 1989.

[27] D. Flotzinger, G. Pfurtscheller, C. Neuper, J. Berger, W. Mohl. Classification of non-averaged EEG data by learning vector quantization and the influence of signal preprocessing. *Med. Biol. Eng. Comp.*, vol. 32, pp.571-576, 1994.

[28] G. Pfurtscheller, D. Flotzinger, M. Pregenzer, J.R. Wolpaw, and D. McFarland. EEG-based Brain Computer Interface (BCI): Search for optimal electrode position and frequency components. *Medical Progress through Technology*, vol. 21, pp. 111-121, 1996.

[29] M. Pregenzer, G. Pfurtscheller, and D. Flotzinger. Automated feature selection with a distinction sensitive learning vector quantizer. *Neurocomputing*, vol. 11, pp. 19-29, 1996.

[30] G.W. Neat, D.J. McFarland, C.A. Forneris, and J.R. Wolpaw. EEG-based brain-to-computer communication: system description. *Proceedings of IEEE EMBS*, vol. 5, pp. 2298-2230, 1990.

[31] D.J. McFarland, A.T. Lefkowicz, and J.R. Wolpaw. Design and operation of an EEG-based brain-computer interface (BCI) with digital signal processing technology. *Behavioral Research Methods, Instuments, and Computers*, vol. 29, pp. 337-345, 1996.

[32] A. Kostov and M. Polak. Parallel man-machine training in development of EEG-based cursor control. *IEEE Transactions on Rehabilitation Engineering*, vol. 8, no. 2, pp. 203-205, 2000.

[33] H.H. Jasper. The Ten-Twenty Electrode System of the International Federation in Electroencephalography and Clinical Neurophysiology. *Electroencephalography and clinical Neurophysiology*, vol. 10, pp. 371-375, 1958.

[34] D.J. McFarland, L.M. McCane, and J.R. Wolpaw. EEG-based communication and

control: Short-term role of feedback. *IEEE Transactions on Rehabilitation Engineering*,, vol. 6, no. 1, pp. 7-11, 1998.

[35] M. Polak and A. Kostov. Feature extraction in development of Brain-Computer Interface: A case study. *Proceedings of IEEE EMBS*, vol. 20, pp. 2058-2061, 1998.

[36] R.O. Duda and P.E. Hart. *Pattern Classification and Scene Analysis*. Wiley-Interscience Publication, 1973.

[37] D.J. McFarland, G.W. Neat, R.F. Read and J.R. Wolpaw. An EEG-based method for graded cursor control. *Psychobiology*, vol. 21, pp. 77-81, 1993.

[38] P.B.C. Fenwick, P. Mitchie, J. Dollimore, and G.W. Fenton. Application of the autoregressive model to EEG analysis. *Agressologie*, vol. 10, pp. 553-564, 1969.

[39] G. Pfurtscheller and G. Haring. The use of an EEG autoregressive model for the time saving calculation of spectral power density distribution with a digital computer. *Electroencephalography and clinical Neurophysiology*, vol. 33, pp. 113-115, 1972.

[40] B.H. Jansen, A. Hasman, R. Lenten, and S.L. Visser. The usefulness of autoregressive models to classify EEG segments. *BioMedizinische Technik*, vol. 24, pp. 216-223, 1979.

[41] M.J. Foster, D.J. McFarland, J.R. Wolpaw. Improvement in EEG-based brain-computer communication by use of additional recording locations. *RESNA'95*, pp.687-689, 1995.

[42] B.H Jansen, J.R. Bourne, and J.W. Ward. Autoregressive estimation of short segment spectra for computerized EEG analysis. *IEEE Transactions on Biomedical Engineering*, vol. 28, pp. 630-638, 1981.

[43] W.H. Press, S.A. Teukolsky, W.T. Vetterling, and B.P. Flannery. *Numerical Recipes in C: The Art of Scientific Computing*. 2nd ed., Cambridge University Press, 1995.

[44] G.E.P. Box, G.M. Jenkins, and G.C. Reinsel. *Time Series Analysis: Forecasting and Control*. 3rd ed., Prentice-Hall, 1994.

[45] M. Laurie, D.J. McFarland, J.R. Wolpaw. Answering questions with an electroencephalogram-based brain-computer interface. *Arch. Phys. Med. Rehabil.*,. Vol. 79, pp. 1029-1033, 1998.

[46] W.W. Armstrong and M.M. Thomas. *Atree 3.0 ALN development system*. Dendronic Decisions Limited, Edmonton, Canada, 1995.

[47] S.G. Mason and G.E. Birch. Processing of single-trial, movement-related ERP's for human-machine interface applications. *RESNA'95*, pp. 673-675, 1995

[48] M.J. Polak, S.H. Zhou, P.M. Rautaharju, W.W. Armstrong, and B.R. Chaitman. Using automated analysis of the resting twelve-lead ECG to identify patients at risk of developing transient myocardial ischemia-an application of a adaptive logic network. *Physiological Measurements*, 1997.

[49] W.W. Armstrong, C. Chu, and M.M. Thomas. Using adaptive logic network to predict machine failure. *World Congress on Neural Networks*, pp. II-80 - II-83, 1995.

[50] A. Kostov, R.B. Stein, W.W. Armstrong, and M.M. Thomas. Evaluation of adaptive logic networks for control of walking in paralyzed patients. *14th Annual International Conference IEEE Engineering in Medicine and Biology Society*, vol.4, pp.1332-1334, 1992.

[51] B.D. Ripley. Pattern Recognition and Neural Networks. *Cambridge University Press*, 1996.

[52] R.A. Fisher. The use of multiple measurements in taxonomic problems. *Annals of Eugenics*, vol. 7, pp.179-188, 1936.

[53] C.A.B. Smith. Some examples of discrimination. *Annals of Eugenics*, vol.13, pp.

272-282, 1947.

[54] V.B. Rao and H.V. Rao. *C++ Neural Networks and Fuzzy Logic*. MIS Press, 1993.

[55] R. Hecht-Nielsen. *Neurocomputing*. Addison-Wesley Publishing, 1990.

[56] T.G. Dietterich. Machine learning research: Four current directions. Draft of May 23, 1997.

[57] L. Hansen and P. Salamon. Neural network ensembles. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 12, pp. 993-1001, 1990.

[58] P.H. Peckham and M.W. Keith. *Motor Prostheses for Restoration of Upper Extremity Function, in Neural prostheses: Replacing Motor Function After Disease or Disability*. Oxford University Press, pp. 162-187, 1992.

[59] Enabling Devices, a division of Toys for Special Children, Inc. Product Information (www.enablingdevices.com)

[60] ACS Technologies, Inc. P.O. Box 889, Jackson, MI, USA 49204, phone (800)788-0889

[61] IntelliTools, Inc., 55 Leveroni Court, Suite 9, Novato, CA, USA 94949 (www.intellitools.com)

[62] Kirkup, L. et al., "EEG-Based system for rapid on-off switching without prior learning", Medical and Biological Engineering and Computing, v35n5 Sep 1997, pp. 504-509.

[63] R.E. Isaac. Real-time Control of a Cortical Neural Prosthesis.

[64] P.R. Kennedy and R.A.E. Bakay. Restoration of neural output from a paralysed patient by a direct brain connection. *NeuroReport*, vol. 9, no. 8, 1998.

[65] L. Breiman. Hinging hyperplanes for regression, classification and function approximation. *IEEE Transactions on Information Theory*, vol. 3, pp. 999-1013, 1993.

[66] T. Kohonen. Learning vector quantization. *Neural Networks*, vol. 1 (suppl. 1), pp. 303, 1988.

Appendix

Figures A-1, A-2, A-3, A-4, A-5, and A-6 show the averaged UP and DOWN power spectra for subjects J.K., L.U., R.J., F.K., G.M., and L.G., respectively, that were recorded during the testing phase of one-dimensional control sessions. The y-axis in each figure shows the power component, and the x-axis shows the frequency. The leads that are plotted are the leads that were used for feedback. The y-axis scale is kept constant for each subject between the different leads.

Figures A-3 and A-6 show classical μ-rhythm components. Figures A-2 and A-5 show some possible muscle artifacts, likely combined with β-rhythms. Subject G.M. (Figure A-5) is suffering from post-polio and has problems remaining motionless when breathing, so there are some artifacts present. Subject L.U. (Figure A-2) shows some unexplained frequency components above 20 Hz in leads P3 and P4. It seems like these components may be a combination of β-rhythms and muscle artifacts. Subject L.U. only stayed for 5 sessions and never achieved good cursor control.

Figures A-7 and A-8 show the averaged power spectra that were recorded during the testing phase of two-dimensional control sessions. The spectrum for subject G.M. (Figure A-7) shows difference between directions of control just below 8 Hz and then in the β-rhythm range. The spectrum for subject L.G. (Figure A-8) indicates that the largest differences between direction of control occurs in the μ-rhythm range.

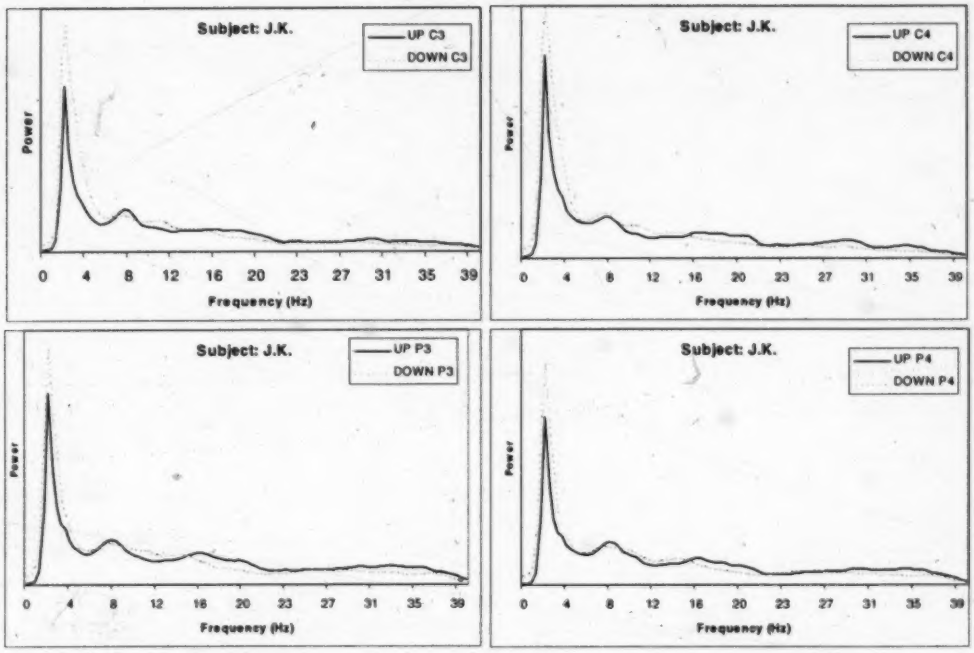

Figure A- 1. Averaged spectra in the testing phase of a session for subject J.K.

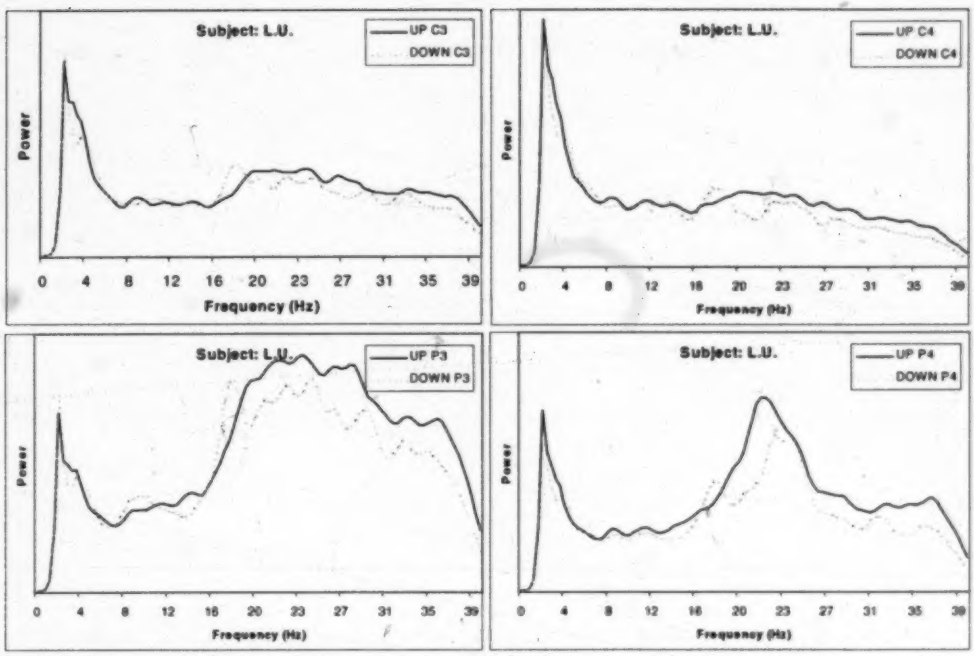

Figure A- 2. Averaged spectra in the testing phase of a session for subject L.U.

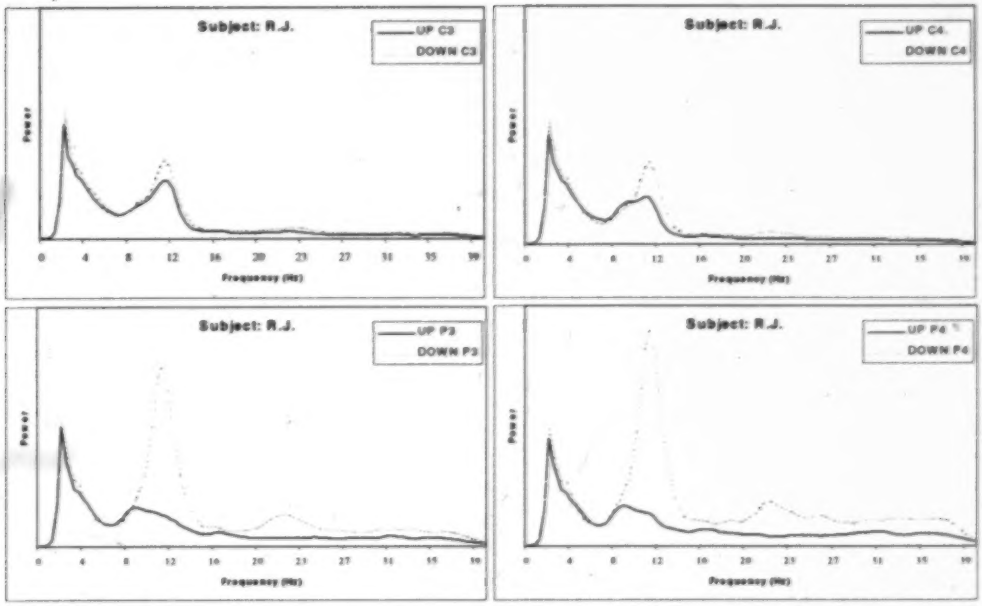

Figure A- 3. Averaged spectra in the testing phase of a session for subject R.J.

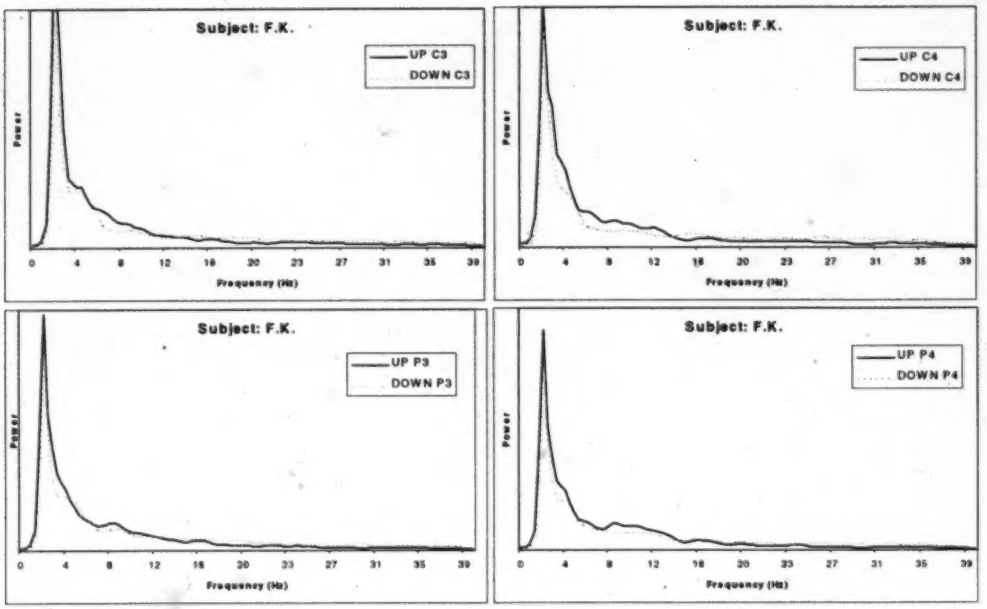

Figure A- 4. Averaged spectra in the testing phase of a session for subject F.K.

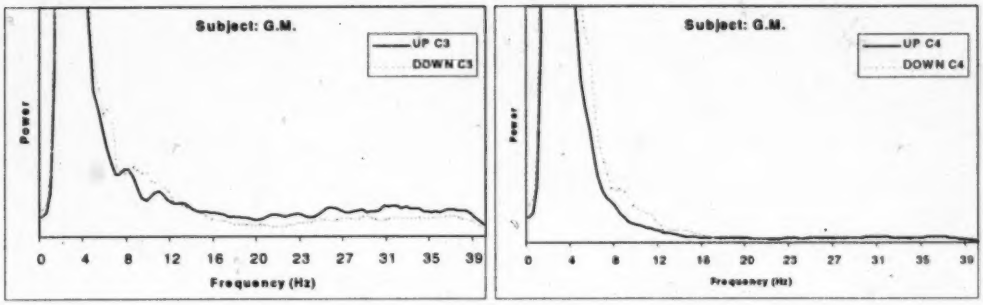

Figure A- 5. Averaged spectra in the testing phase of a session for subject G.M.

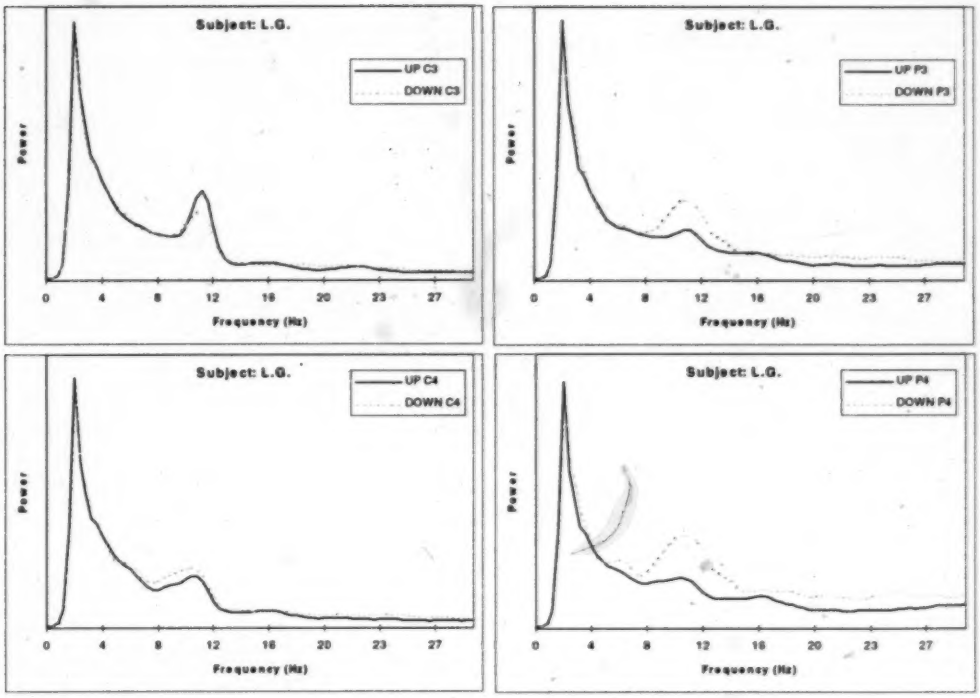

Figure A- 6. Averaged spectra in the testing phase of a session for subject L.G.

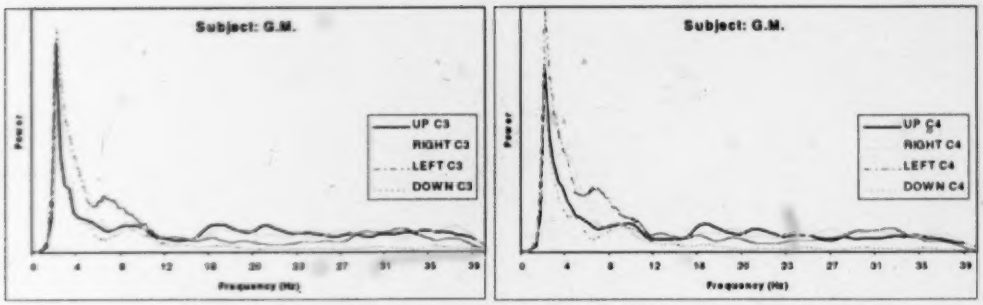

Figure A-7. Averaged spectra in the 2-dimensional testing phase of a session for subject G.M.

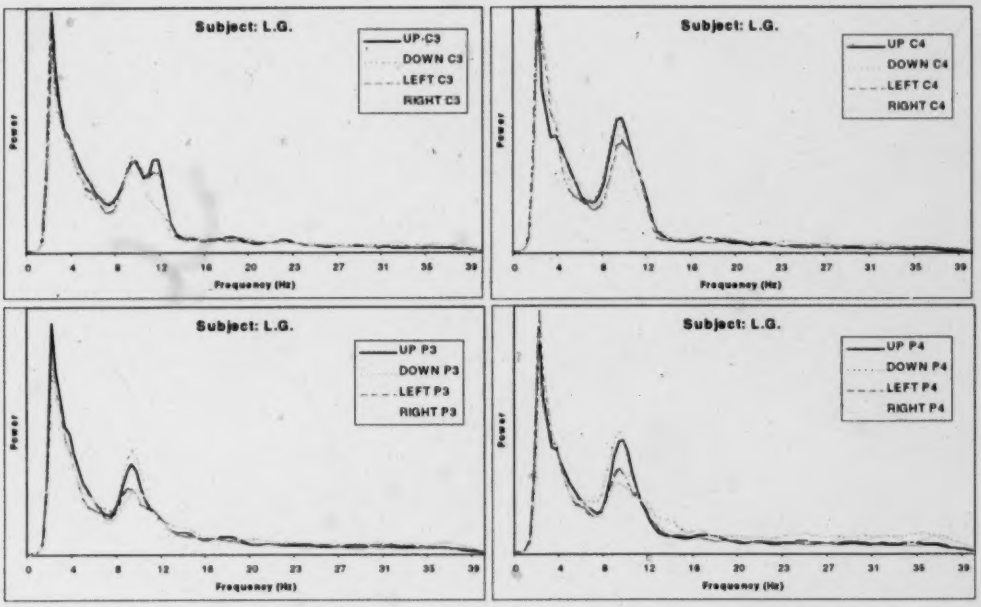

Figure A-8. Averaged spectra in the 2-dimensional testing phase of a session for subject L.G.